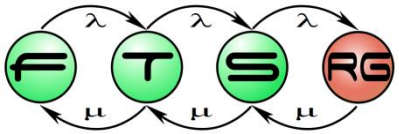
Leistungsmodellierung

Budapest University of Technology and Economics
Fault Tolerant Systems Research Group



Ein bekanntes Beispiel …

Ein bekanntes Beispiel …

Kiszolgálóhiba történt az alkalmazásban: „/hallgato”.

Futásidejű hiba

Leírás: Alkalmazáshiba történt a kiszolgálón. Az alkalmazás jelenlegi egyéni hibakezelési beállításai (biztonsági okok
es adatainak távoli megjelenítését. A
böngészőben azonban megtekinthetők.

Részletek: Ha távoli gépeken is meg szeretné jeleníteni a hibaüzenet részletes adatait, hozzon lé
ában elhelyezett „web.config”
címke „mode” attribútumát állítsa „Off” értékére.

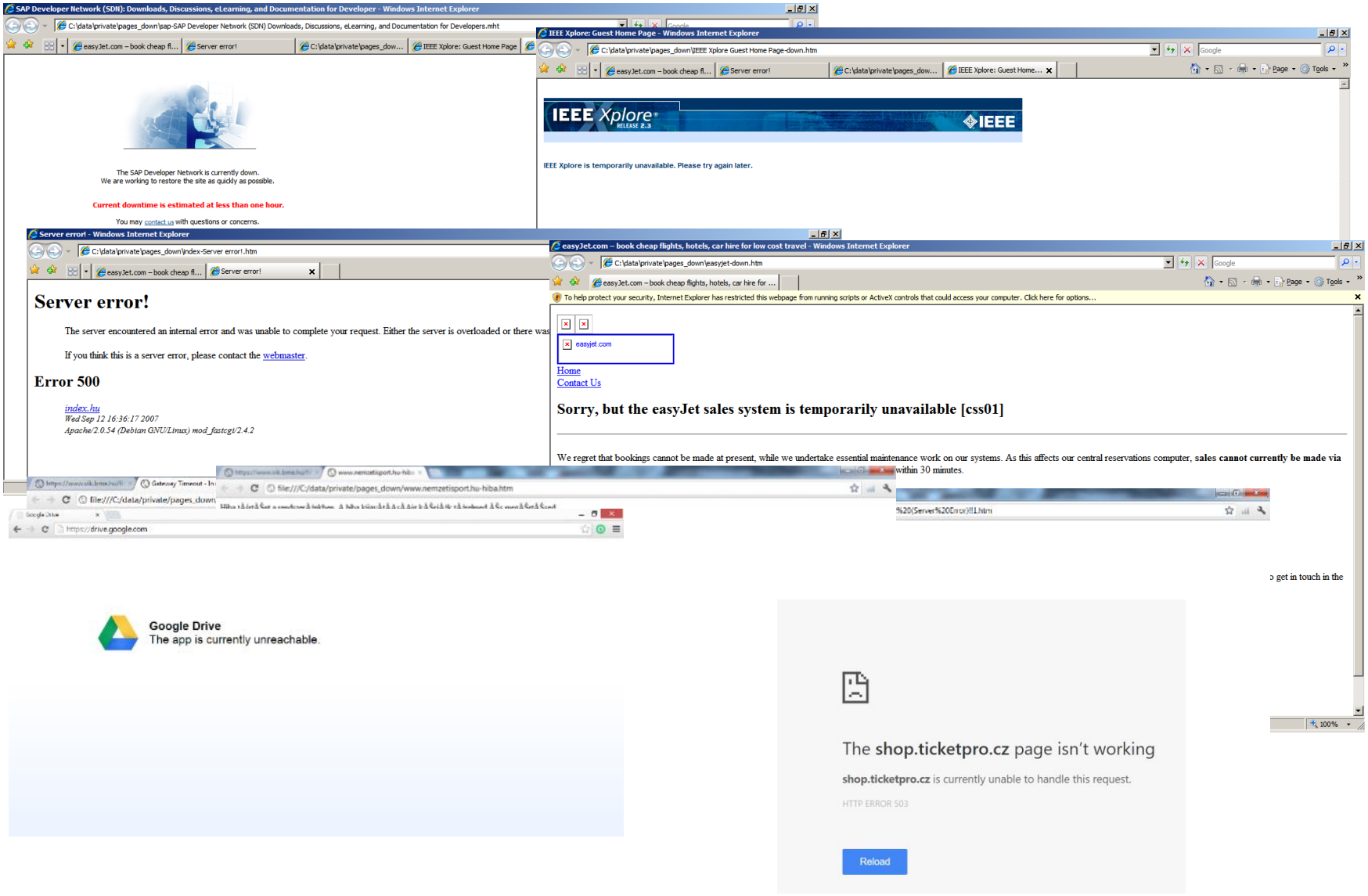
```
<!-- Web.Config konfigurációs fájl -->

<configuration>
    <system.web>
        <customErrors mode="Off"/>
    </system.web>
</configuration>
```

Megjegyzések:

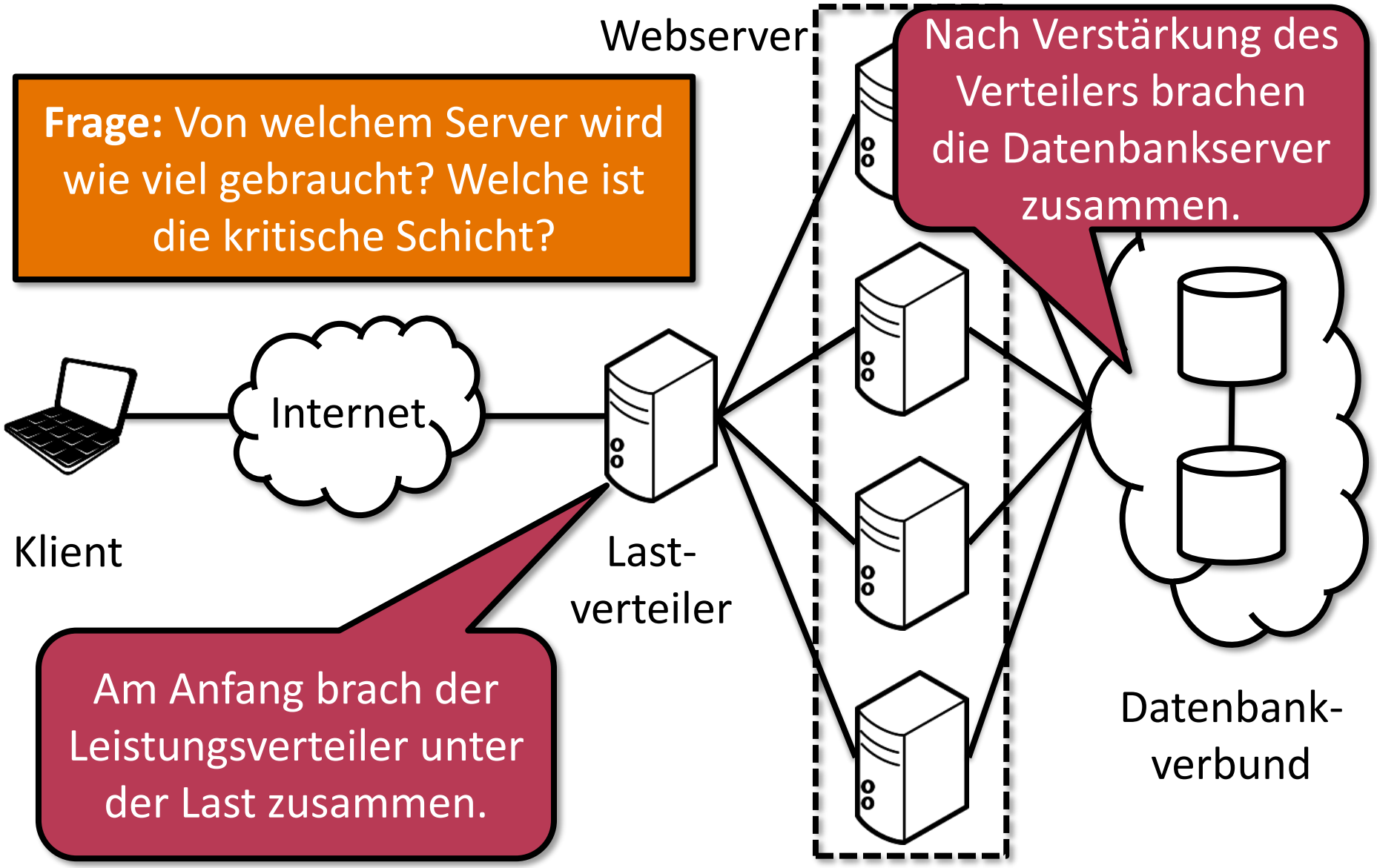
```
<!-- Web.Config

<configuration>
    <system.web>
        <customErr
    </system.web>
</configuration>
```

Motivation: Die Vorbereitung auf die Belastung beginnt während des Entwurfs!

… Und weitere Beispiele

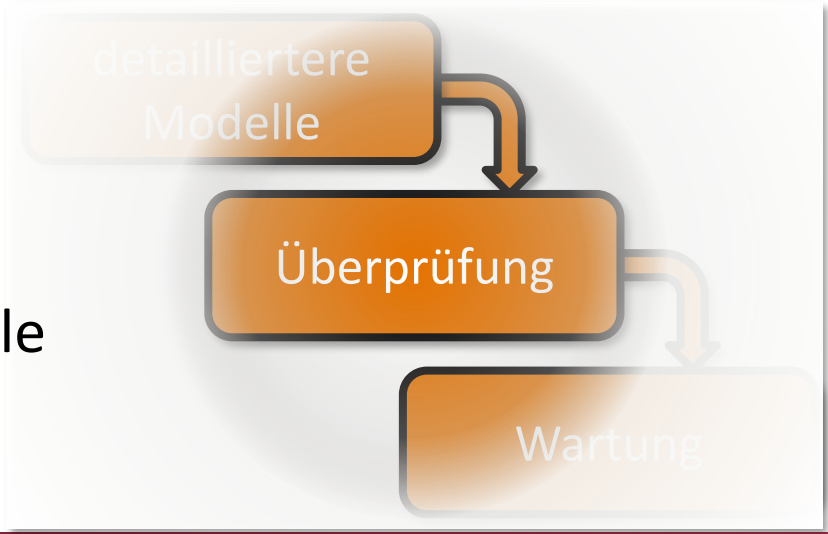


Schematischer Aufbau des Neptun-Systems



Leistungsmodellierung

- Zur Erinnerung: nichtfunktionale Anforderungen
  - Leistung, maximaler Durchsatz, Zuverlässlichkeit, etc.
  - Wie können sie (vor der Fertigstellung) überprüft werden?
- **Leistungsmodellierung:**
  - Ergänzung der bisherigen Modelle mit Zeit, Ressourcen, Kapazitätsgrenzen, …
  - Ziel:
    - Bewertung der Systemleistung in der Entwurfsphase
    - Identifikation der kritischen Teile
    - Skalierung, Dimensionierung



Inhalt

Grundbegriffe

Belastungsdiagramme

Ressourcemodellierung

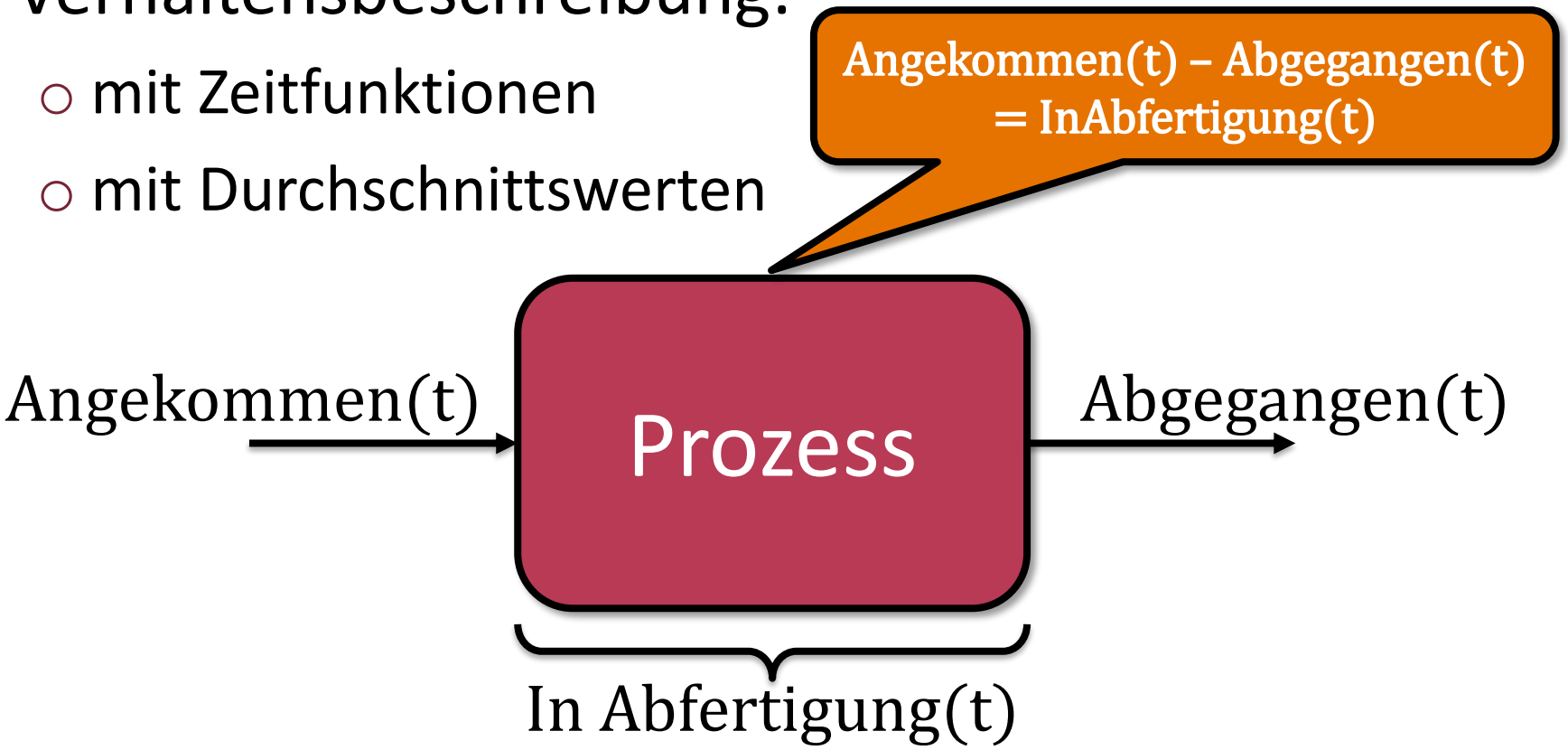
Das Ziel der Leistungsmodellierung

- Kein Ziel:
  - Ausführliche Modellierung der internen Operation des Systems (Dies wäre in der Praxis unmöglich.)
  - Beschreibung genauer Zusammenhänge
- **Ziel:**
  - Schwachstellen zu identifizieren (für Weiterentwicklung)
  - die Arbeitsmodi des Systems zu bestimmen
  - die **Natur** der Zusammenhänge zu beschreiben
  - Definition der Metriken
    - die für die verschiedenen Teilnehmer/Rollen sichtbar sind

GRUNDBEGRIFFE

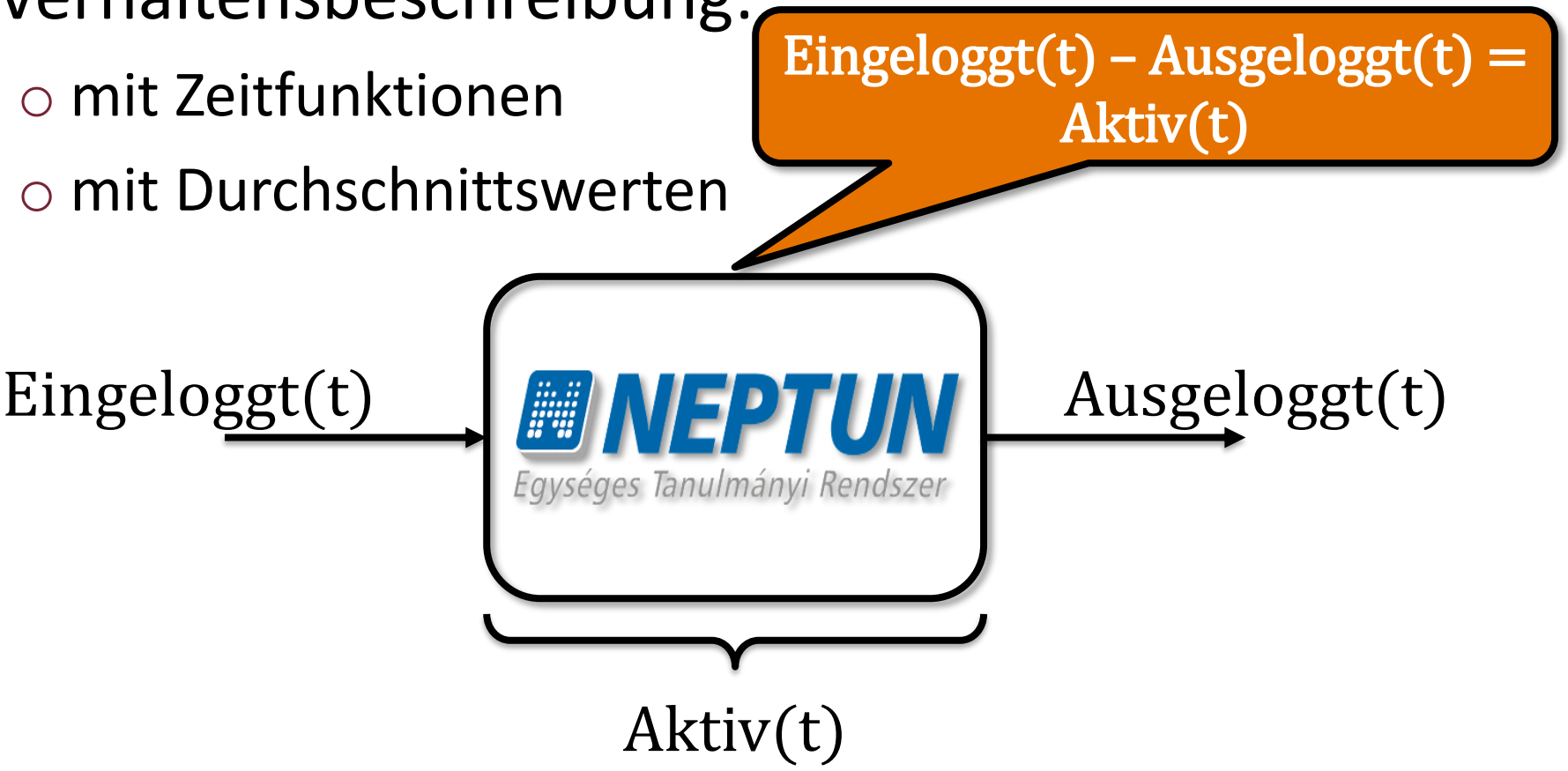
Das Grundmodell

- Ausführung eines Prozesses für mehrere Anfragen
  - Untersuchungsgegenstand: zeitabhängiges Verhalten
- Verhaltensbeschreibung:
  - mit Zeitfunktionen
  - mit Durchschnittswerten



Das Grundmodell

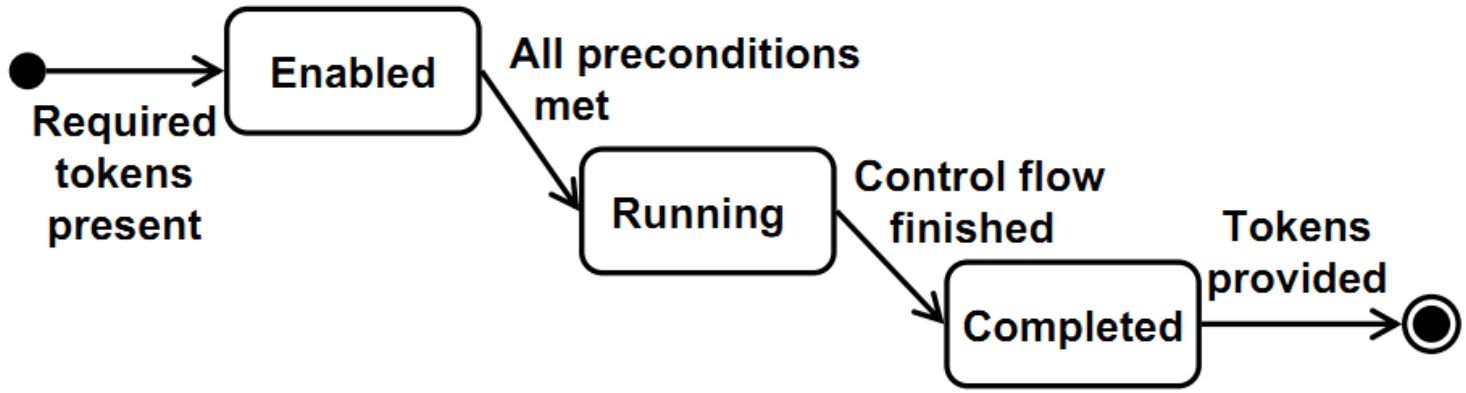
- Ausführung eines Prozesses für mehrere Anfragen
  - Untersuchungsgegenstand: zeitabhängiges Verhalten
- Verhaltensbeschreibung:
  - mit Zeitfunktionen
  - mit Durchschnittswerten



Zur Erinnerung: Zustände der Prozesse

- Zustände der Prozessausführung:

State Machine



- Angekommen(t): Anzahl der Token in „*Enabled*" Zustand
- In Abfertigung(t): Anzahl der Token in „*Running*" Zustand
- Abgegangen(t): Anzahl der Token in „*Completed*" Zustand

Definition: Ankunftsrate, Durchsatz

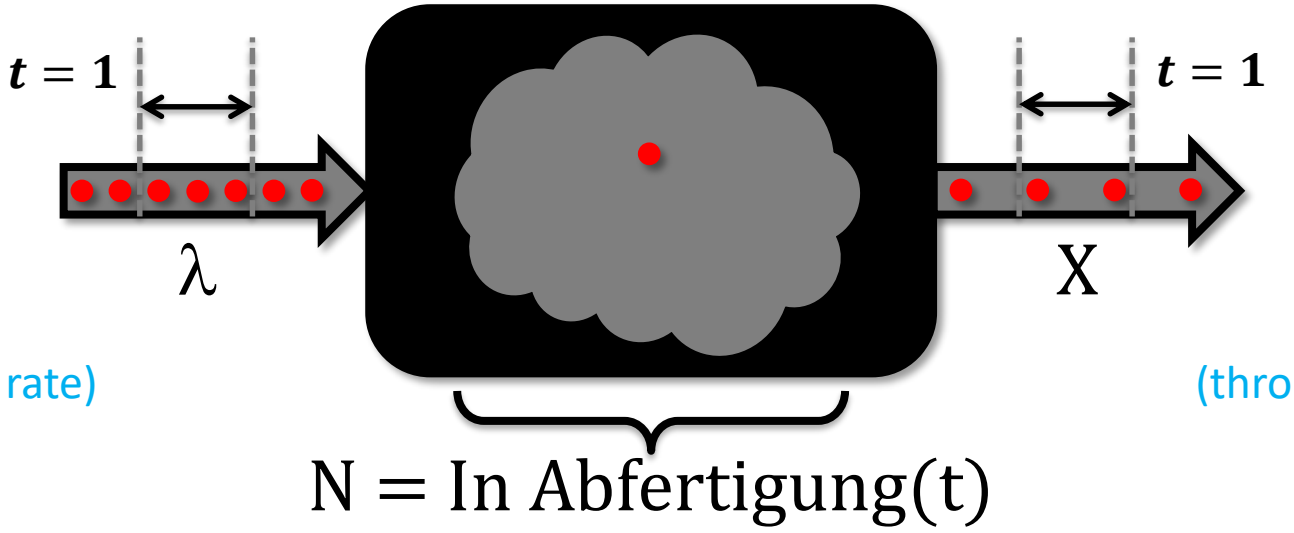
Ankunftsrate: Anzahl der pro Zeiteinheit angekommenen Anfragen.

$$\lambda = \frac{Angekommen\ (t)}{t} \qquad [\lambda] = \frac{1}{s}$$

Durchsatz: Anzahl der pro Zeiteinheit abgefertigten Anfragen.

$$X = \frac{Abgegangen(t)}{t} \qquad [X] = \frac{1}{s}$$

(auch Bedienrate genannt)



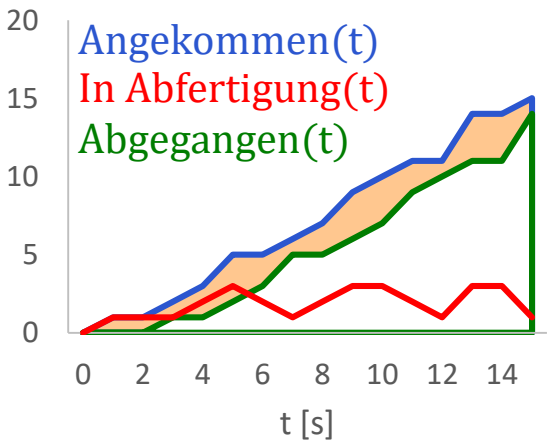
(arrival rate)

(throughput, XPUT)

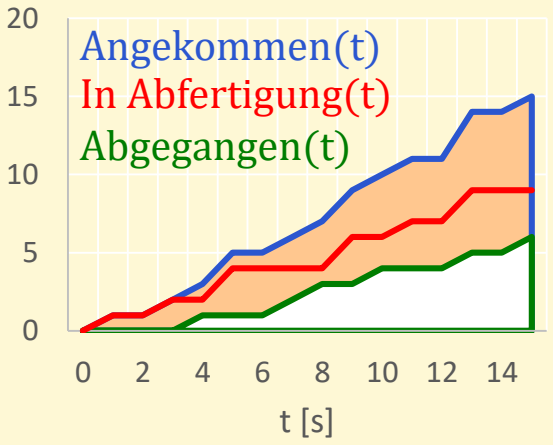
Definition: Gleichgewicht

- **Stabiler Zustand:** *In Abferigung(t) ist beinahe konstant*
  - In diesem Fall kann mit Durchschnittswerten gerechnet werden!
  - Gleichgewicht ist, wenn: $\lambda = X$

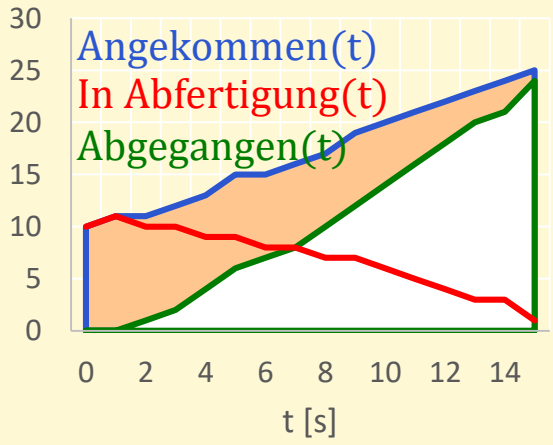
$\lambda = X$



$\lambda > X$



$\lambda < X$



Definition: Gleichgewicht

- **Stabiler Zustand:** *In Abferigung(t) ist beinahe konstant*
  - In diesem Fall kann mit Durchschnittswerten gerechnet werden!
  - Gleichgewicht ist, wenn: $\lambda = X$

Gleichgewicht: Gleich viele Ein- und Austritte pro Minute

Begrenzte Kapazität – DoS

- Bisher konnte N sogar unendlich sein
- Was passiert, wenn sie endlich ist?

Denial of Service Attack

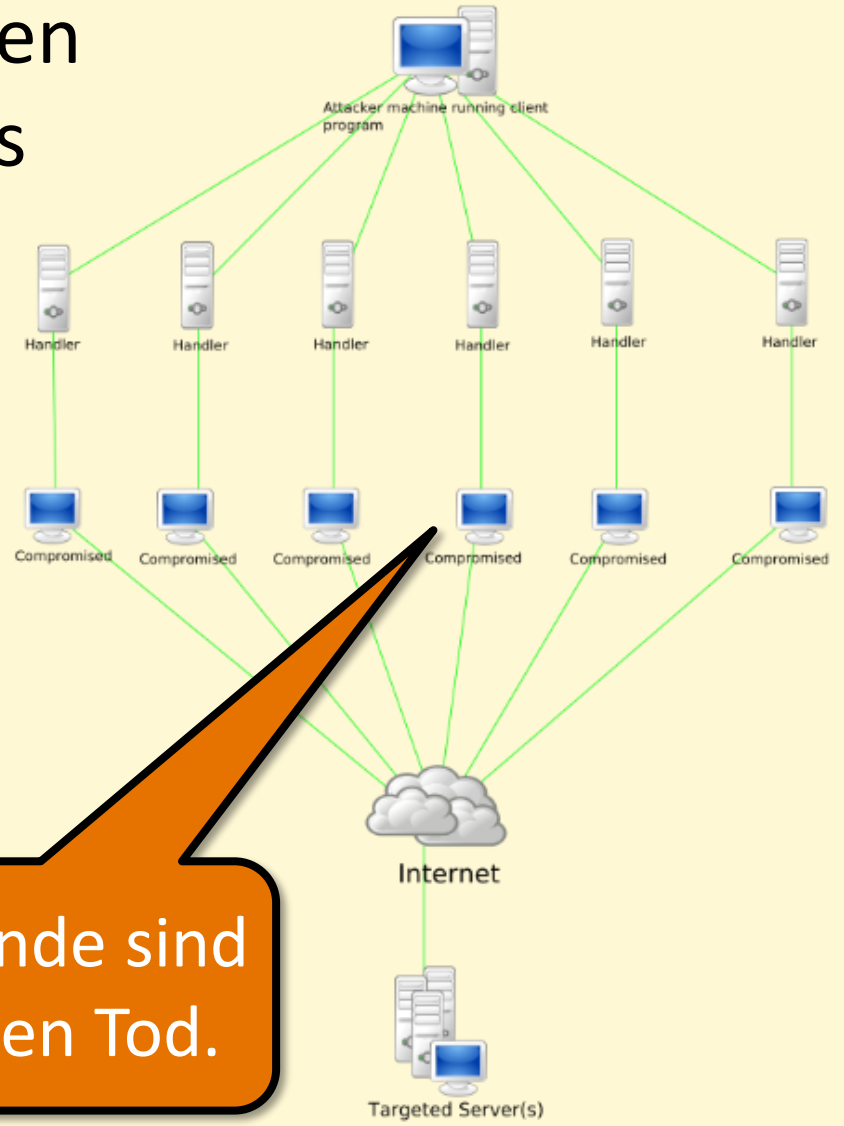
Thursday, August 6, 2009 | By Biz Stone (@biz) 08/06/2009 - 15:00

Tweet

On this otherwise happy Thursday morning, Twitter is the target of a denial of service attack. Attacks such as this are malicious efforts orchestrated to disrupt and make unavailable services such as online banks, credit card payment gateways, and in this case, Twitter for intended customers or users. We are defending against this attack now and will continue to update our status blog as we continue to defend and later investigate.

(Distributed) Denial of Service – (D)DoS

- (Generierte) Massenanfragen
  → Überlastung des Systems
- Überlastete Systeme sind anfälliger
- Totale Abschaltung von Dienstleistungen
- Beliebte Methode des Anonymous-Gruppe

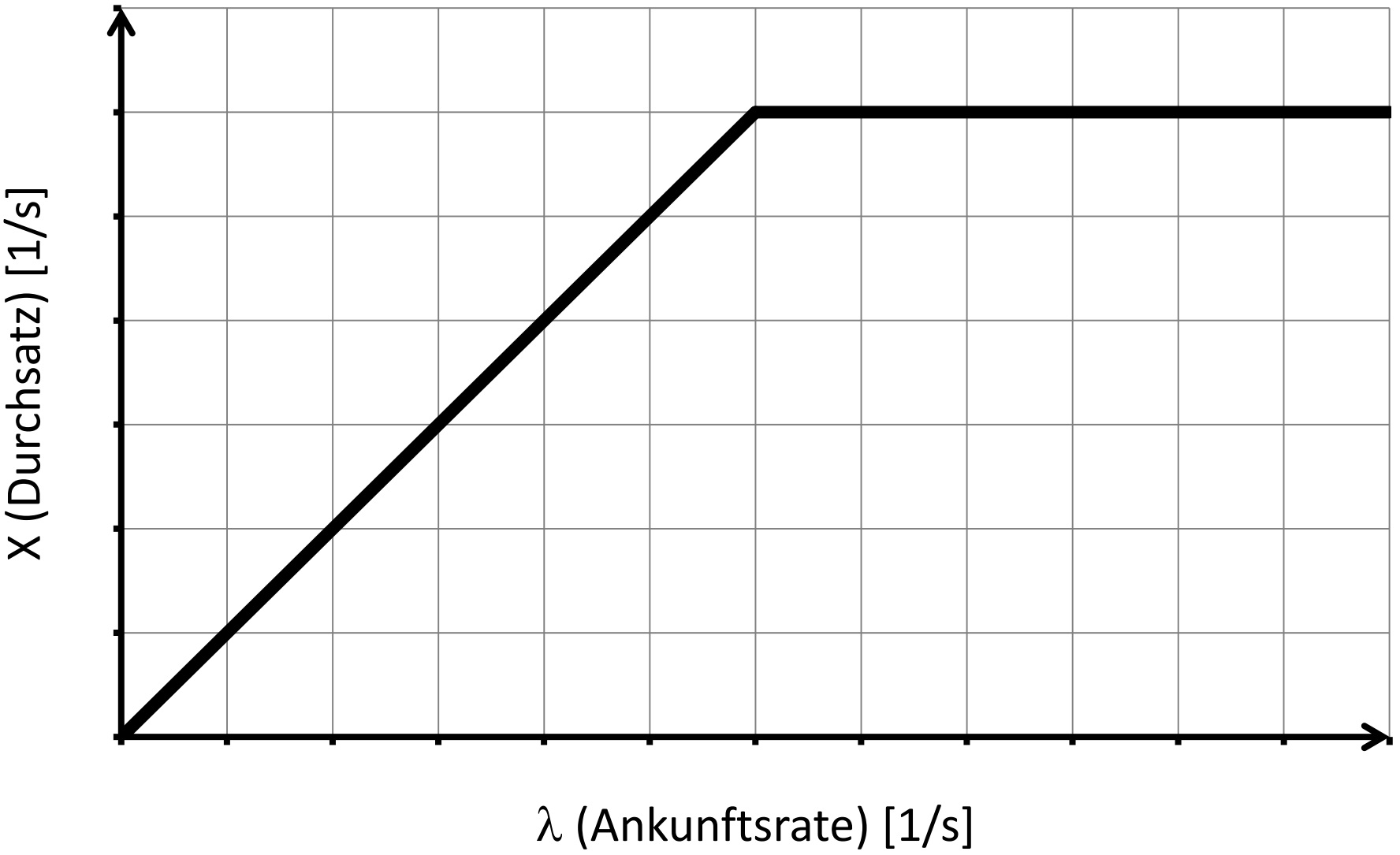


Viele Hunde sind des Hasen Tod.

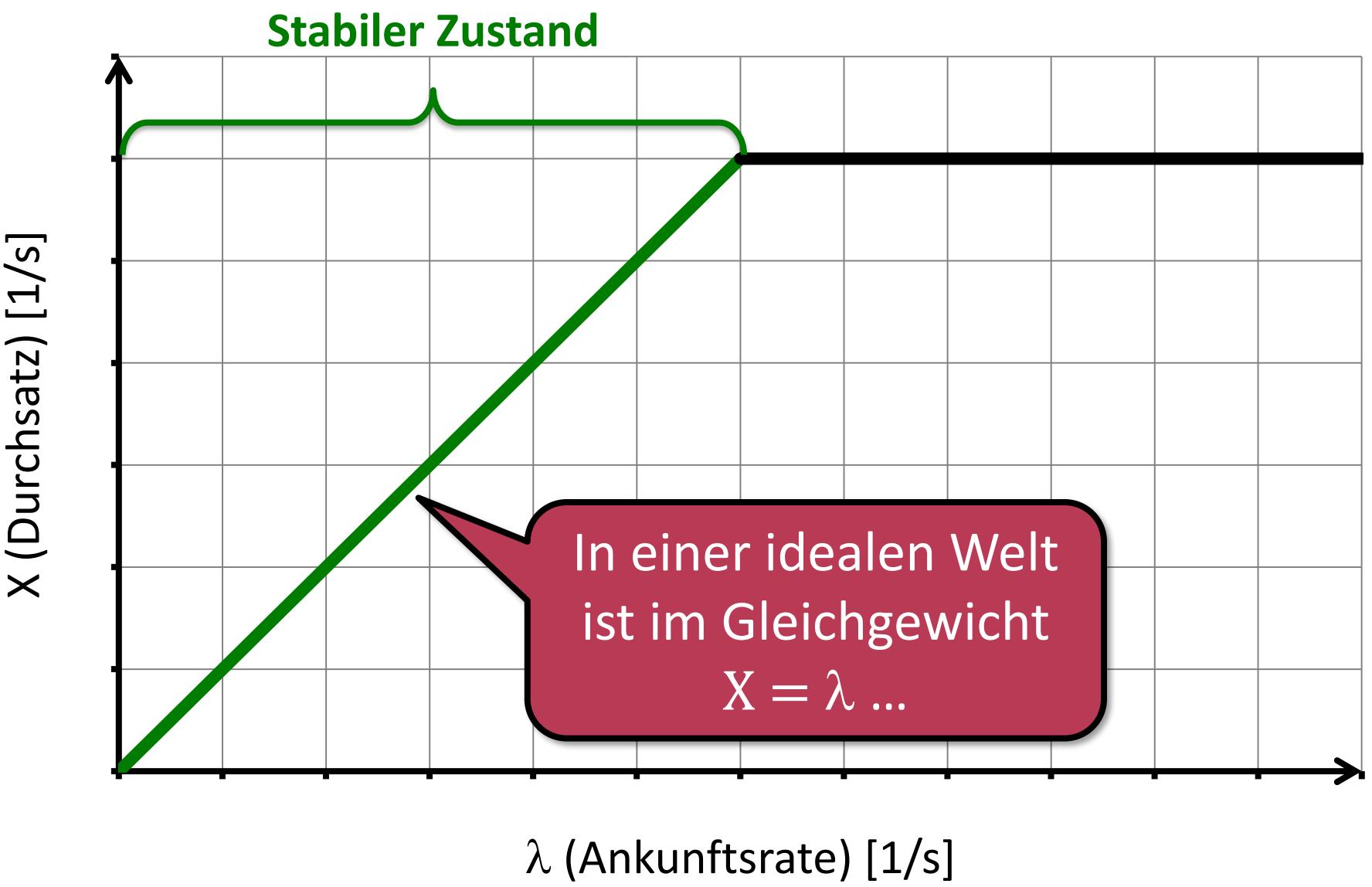
BELASTUNGSDIAGRAMME

- Zusammenhang zwischen Ankuftsrate und Durchsatz
- Grenzdurchsatz

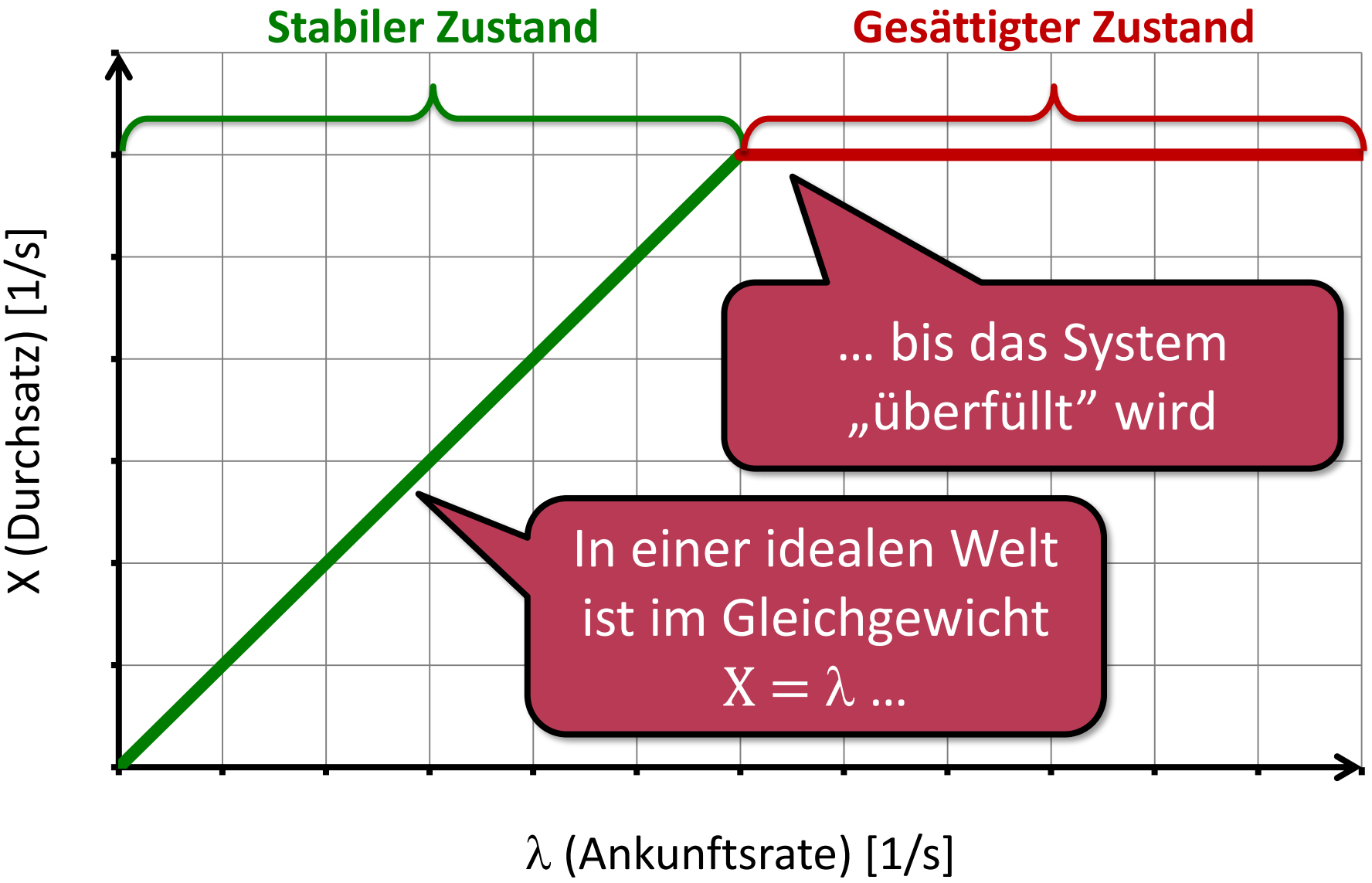
Belastungsdiagramm



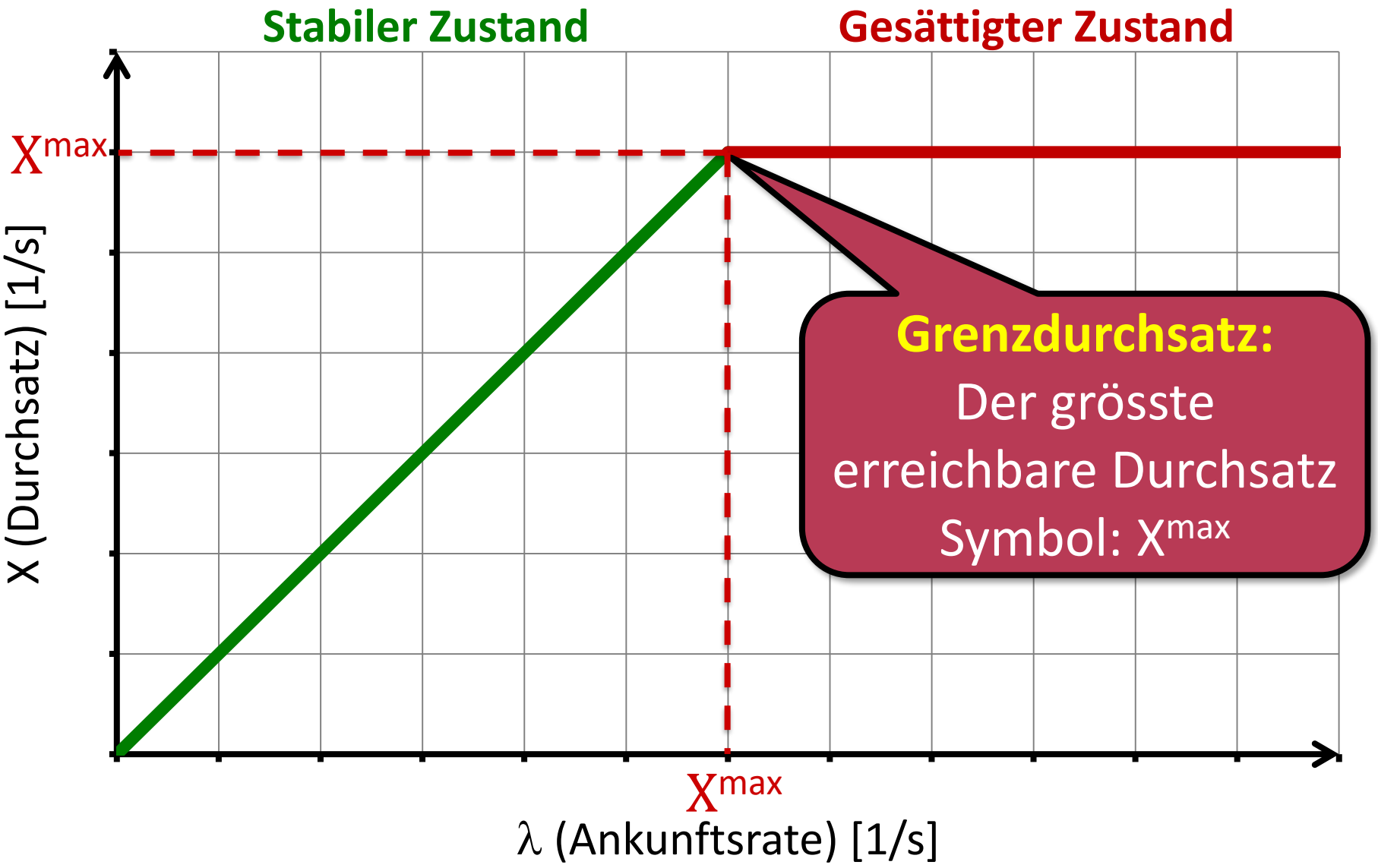
Belastungsdiagramm



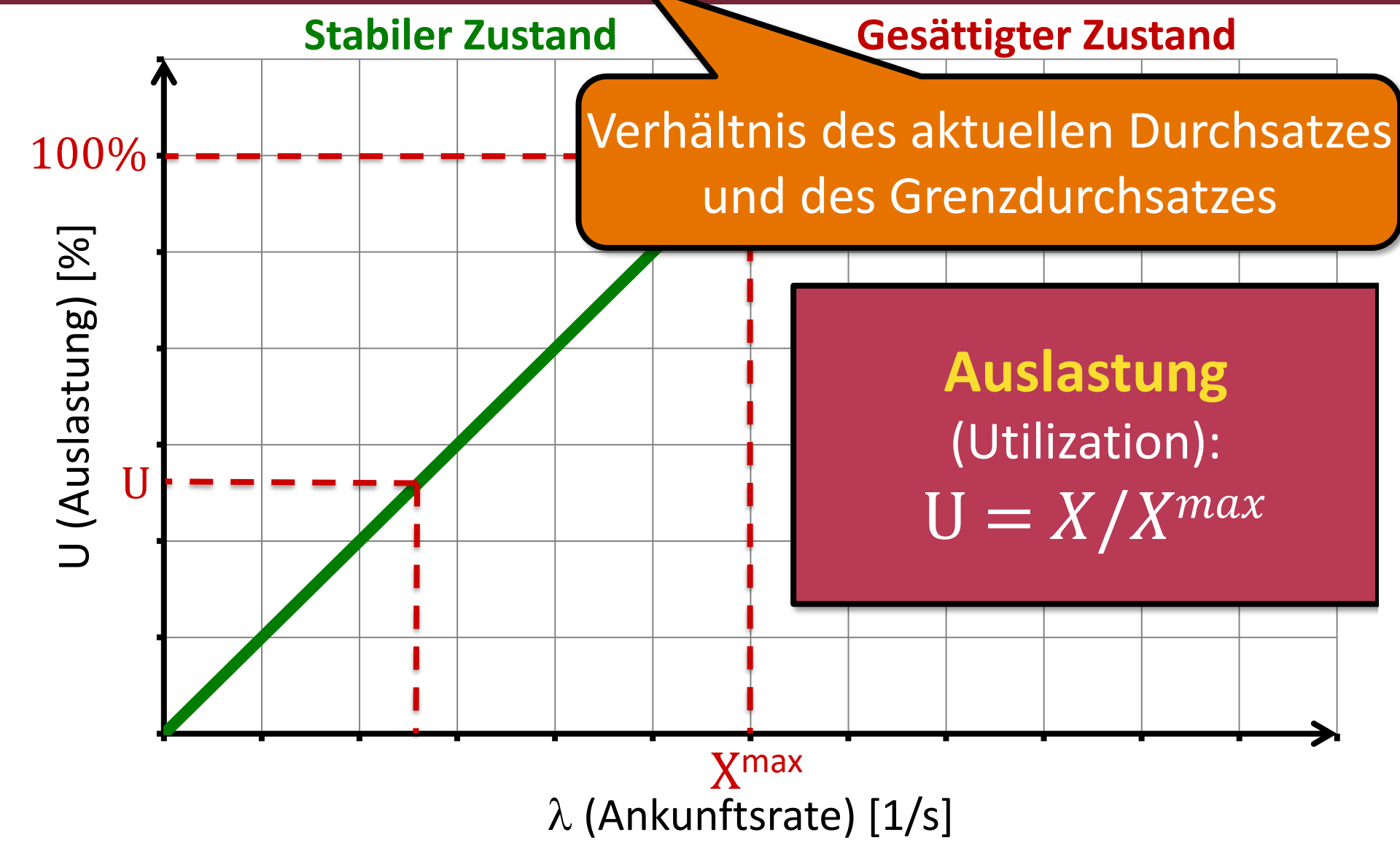
Belastungsdiagramm



Grenzdurchsatz

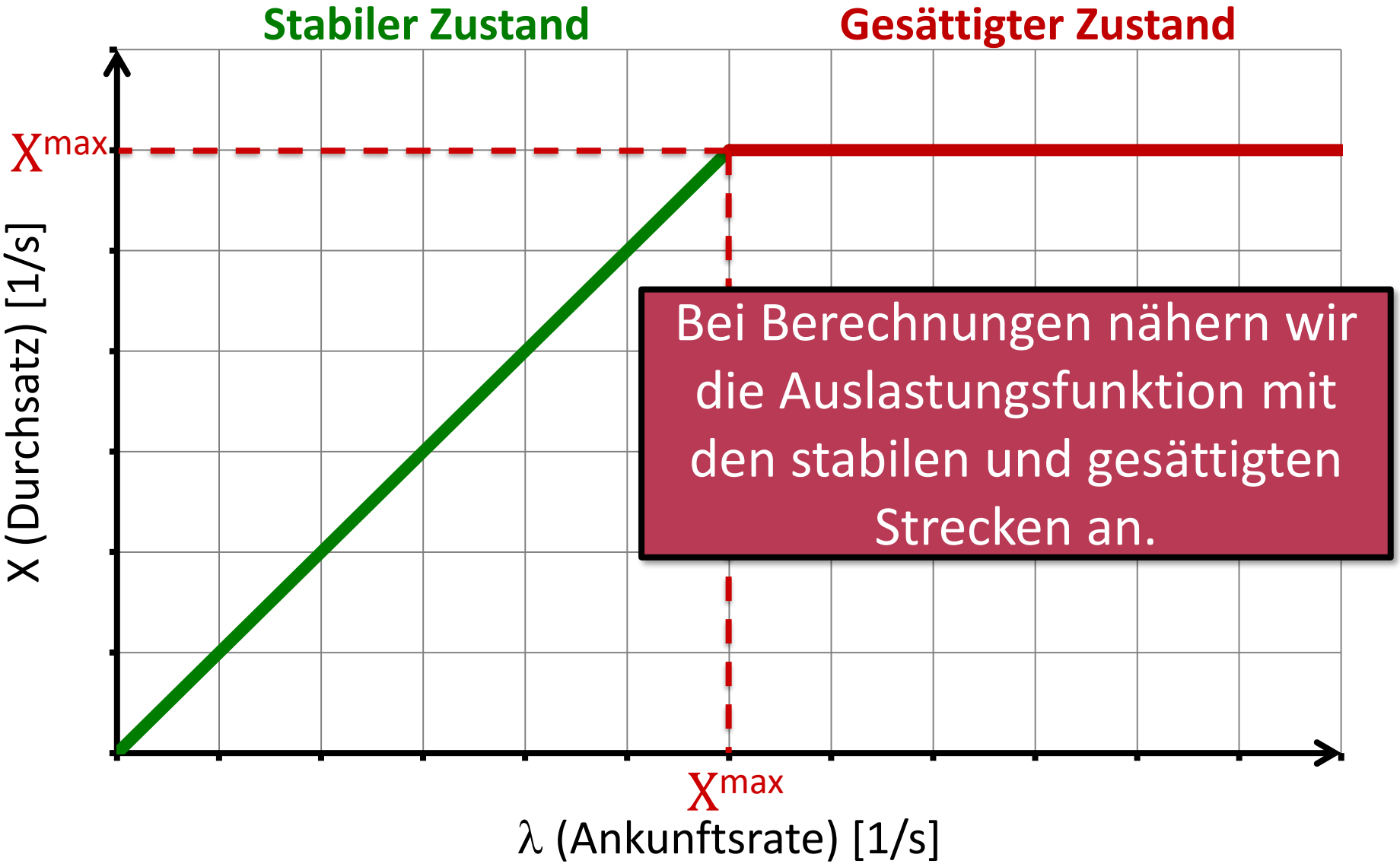


Auslastung



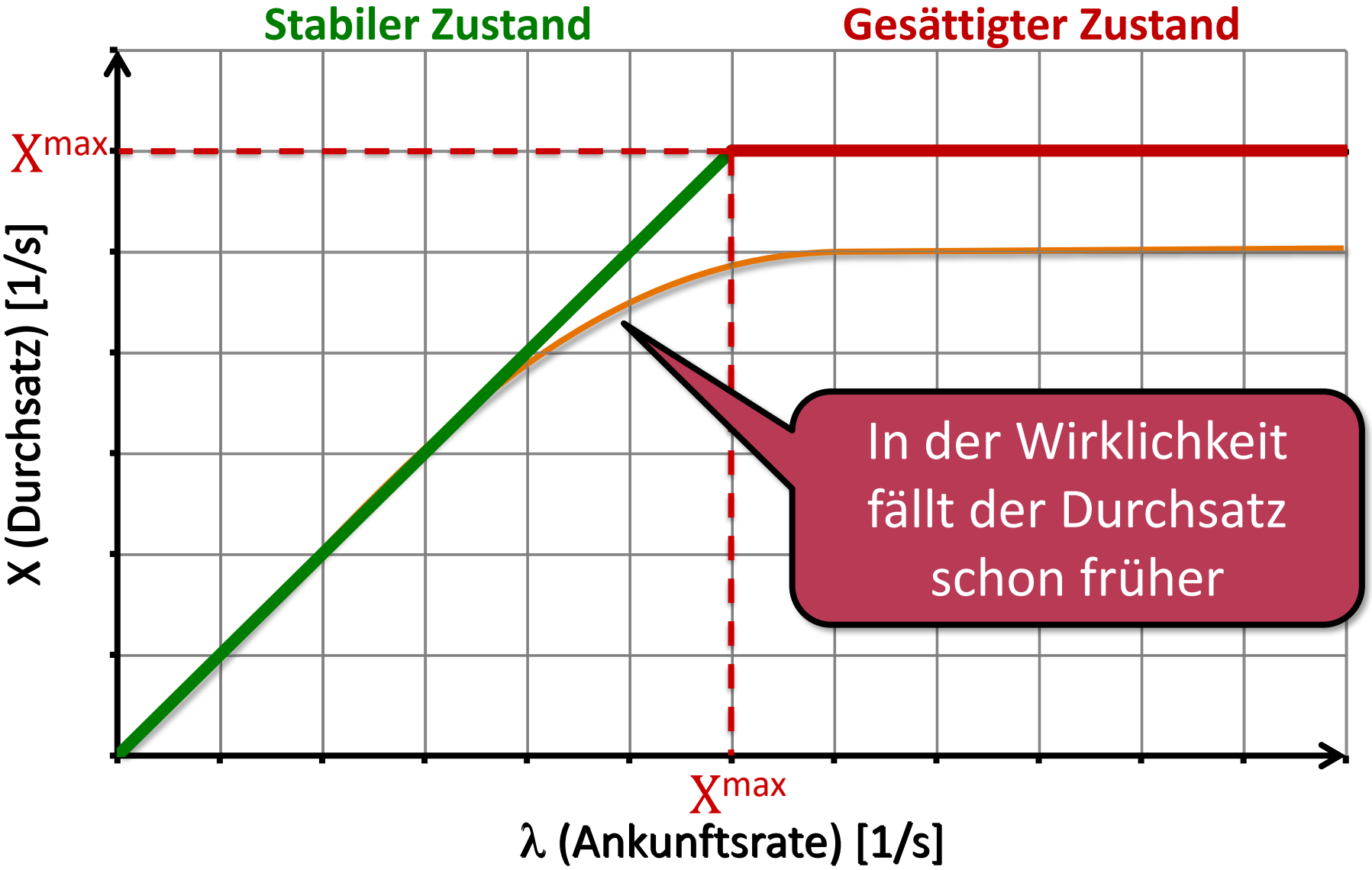
Auslastung
(Utilization):
$U = X / X^{max}$

Näherung der Auslatungsfunktion

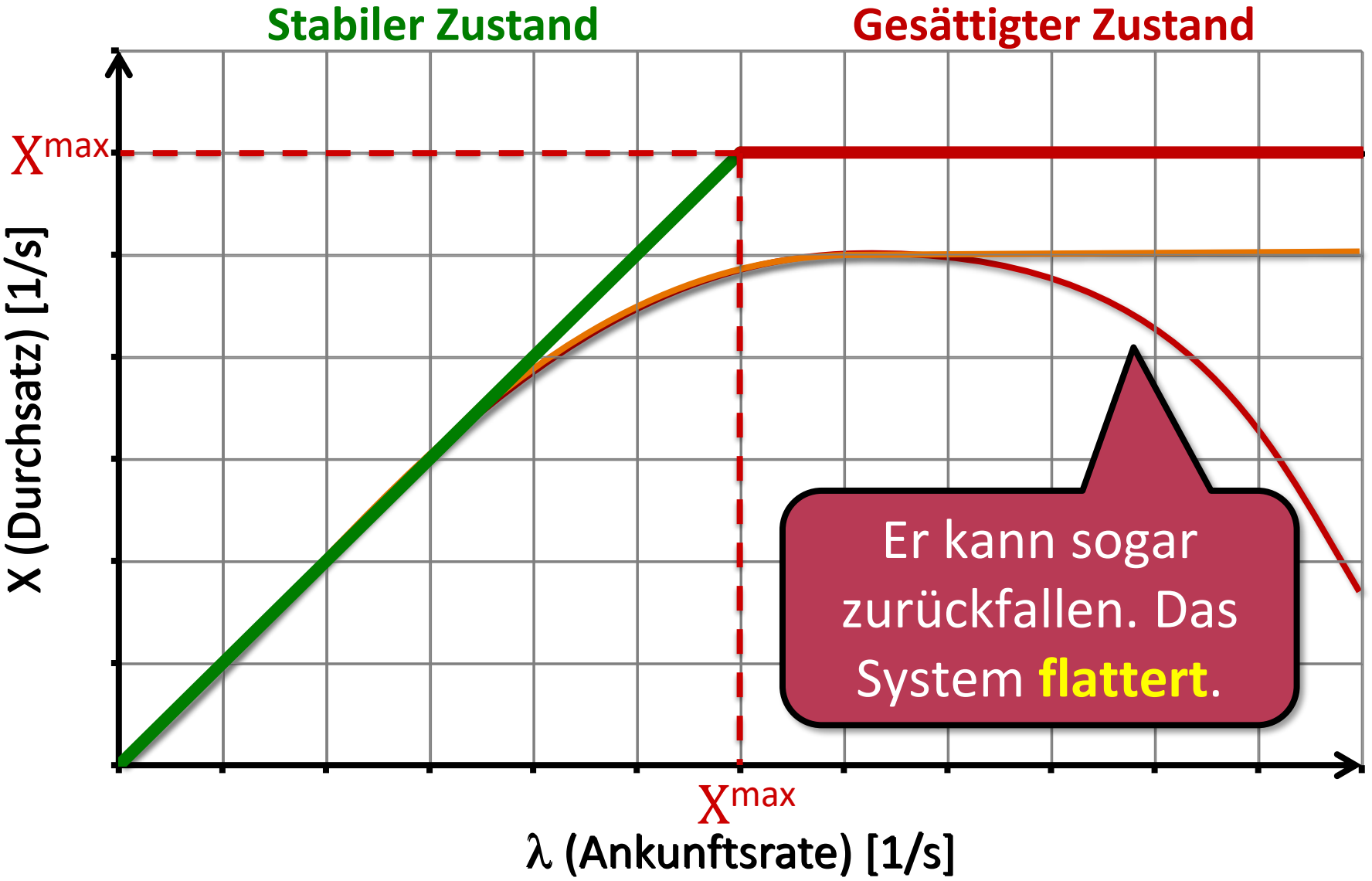


Bei Berechnungen nähern wir die Auslastungsfunktion mit den stabilen und gesättigten Strecken an.

Auslatungsfunktion in der Praxis



Auslatungsfunktion in der Praxis



Definitionen

Grenzdurchsatz: der **grösste erreichbare** Durchsatz

- Symbol: X^{max} (Maximal Throughput)

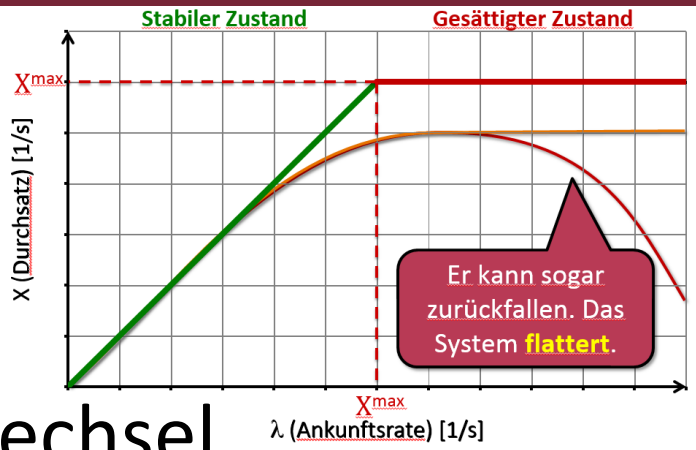
Auslastung: **Verhältnis** des aktuellen und des Grenzdurchsatzes

- Symbol: $U = \frac{X}{X^{max}}$ (Utilisation)

Flattern: in gesättigtem Zustand fällt der Durchsatz zurück

(Thrashing)

Im Modell vernachlässigte Effekte



- Rechenbedarf der Taskwechsel
  - Aufräumen nach dem vorigen
  - Vorbereitung der nächsten
- Rechenbedarf der Ressourcenwechsel
- Mehrfache Sättigung
  - Gleichzeitige Sättigung mehrerer Ressourcen/Server
  - z.B. wenn die M7 staut, staut die Landstraße 7 auch

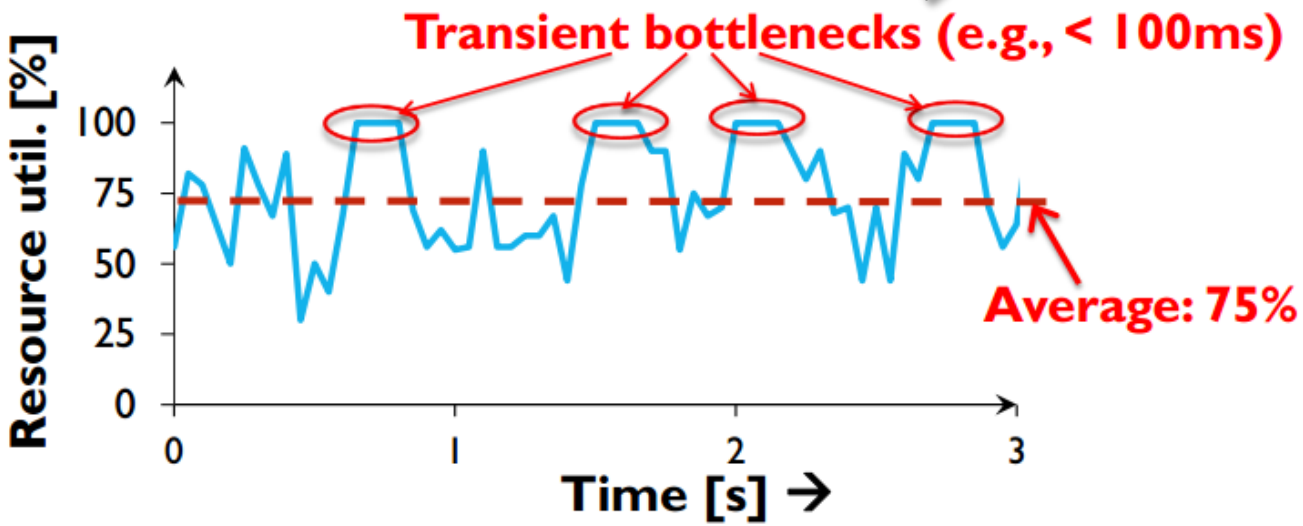
Abgeschreckte Anfragen

- Die Anfragen sind nur bei fanatischen Klienten unabhängig von der Länge der Warteschlange
- Interne Quereffekte im System
  - z.B. unsichtbare Abhängig-keiten unter den Ressourcen
- Offene vs. Zurückgekoppelte Systeme

Effekte der Schwankung der Belastung

- **Durchschnittswerte** vs. **Tatsächliche Belastung**

In gesättigtem System kann die Antwortzeit um 2-3 Grössenordnungen länger sein!



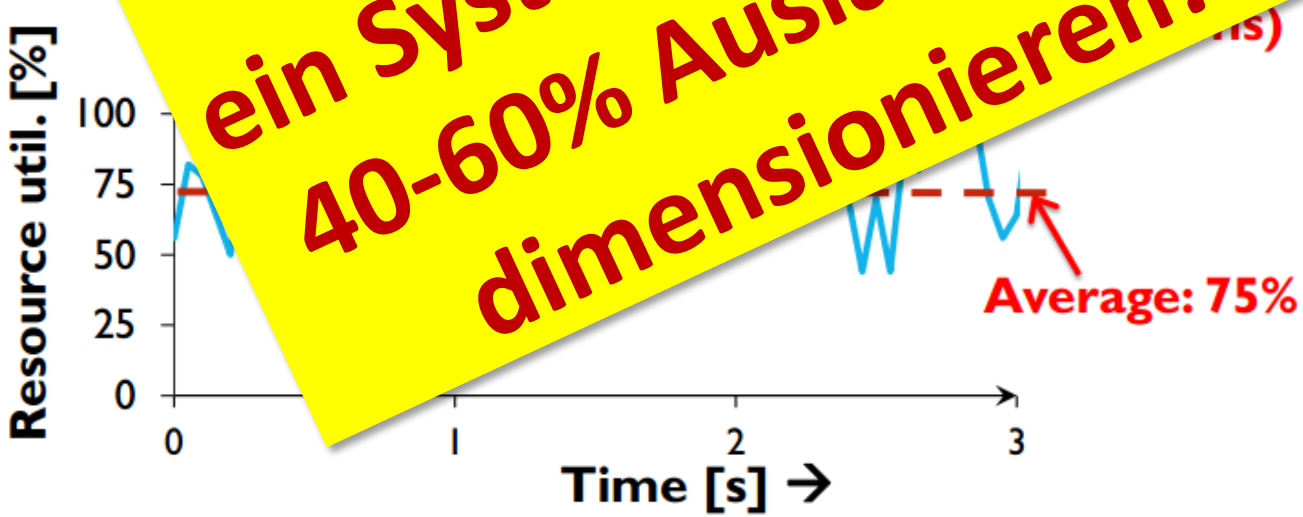
Qingyang Wang, Yasuhiko Kanemasa, Jack Li, Deepal Jayasinghe, Toshihiro Shimizu, Masazumi Matsubara, Motoyuki Kawaba, Calton Pu, "Detecting Transient Bottlenecks in n-Tier Applications through Fine-Grained Analysis", In Proc. of the 33rd International Conference on Distributed Computing Systems (**ICDCS'13**), Philadelphia, Pennsylvania, July 2013.

Effekte der Schwankung der Belastung

- **Durchschnittswerte** vs. **Tatsächliche B lastung**

In g m System
rtzeit um
ungen

Es lohnt sich nicht, ein System für mehr als 40-60% Auslastung zu dimensionieren!



Qingyang Wang, Yasuhiko Kanemasa, Jack Li, Deepal Jayasinghe, Toshihiro Shimizu, Masazumi Matsubara, Motoyuki Kawaba, Calton Pu, "Detecting Transient Bottlenecks in n-Tier Applications through Fine-Grained Analysis", In Proc. of the 33rd International Conference on Distributed Computing Systems (**ICDCS'13**), Philadelphia, Pennsylvania, July 2013.

Neptun – Anmeldung für LVAs

Max. gleichzeitiger Benutzer

Max. gleichzeitiger Benutzer bei optimalem Betrieb

Az elmúlt időkben előfo[illegible]has[illegible]

Dátum	Jelleg	Max. felhasználó	Op. felhasználó
2010.11.29 18:00	vizsga	7303	4623
2010.12.22 06:00	tárgy (EO)	831	831
2011.01.10 18:00	tárgy	12062	4837
2011.01.12 18:00	tárgy (EP)	1765	1765
2011.01.31 16:00	tárgy	1519	1519
2011.05.02 18:00	vizsga	2761	2761
2011.06.07 18:00	tárgy	6095	6095
2011.11.28 18:00	vizsga	4897	4897
2012.01.16 18:00	tárgy	8120	5328
2012.01.30 16:00	tárgy	1703	1703
2012.05.02 18:00	vizsga	2603	2603

Neptun – Anmeldung für LVAs

≈ Ankunftsrate (λ)

≈ Grenzdurchsatz (X_{max})

Az elmúlt időkben előfo[illegible]használósz[illegible]

Dátum	Jelleg	Max. felhasználó	Op. felhasználó
2010.11.29 18:00	vizsga	7303	4623
2010.12.22 06:00	tárgy (EO)	831	831
2011.01.10 18:00	tárgy	12062	4837
2011.01.12 18:00	tárgy (EP)	1765	1765
2011.01.31 16:00	tárgy	1519	1519
2011.05.02 18:00	vizsga	2761	2761
2011.06.07 18:00	tárgy	6095	6095
2011.11.28 18:00	vizsga	4897	4897
2012.01.16 18:00	tárgy	8120	5328
2012.01.30 16:00	tárgy	1703	1703
2012.05.02 18:00	vizsga	2603	2603

Wann war das Neptun-System gesättigt (überlastet)?

Neptun – Anmeldung für LVAs

≈ Ankunftsrate (λ)

atz (X_{max})

Az elmúlt időkben előfo

Dátum	Jelleg		
2010.11.29 18:00	vizsga		
2010.12.22 06:00	tárgy (		
2011.01.10 18:00			
2011.01.12 18:00			
2011.01.31			
2011			2761
2011.			6095
2011.11			4897
2012.01.			5328
2012.01.30		1703	1703
2012.05.02 1		2603	2603

Die Leistung eines überlasteten Systems kann drastisch fallen, es kann sogar zusammenbrechen!

war das Neptun-System gesättigt (überlastet)?

Neptun – Anmeldung für LVAs

≈ Ankunftsrate (λ)

≈ Grenzdurchsatz (X_{max})

Az elmúlt időkben előfo… …használósz…

Dátum	Jelleg	Max. felhasználó	Op. felhasználó
2010.11.29 18:00	vizsga	7303	4623
2010.12.22 06:00	tárgy (EO)	831	831
2011.01.10 18:00	tárgy	12062	4837
2011.01.12 18:00	tárgy (EP)	1765	1765
2011.01.31 16:00	tárgy	1519	1519
2011.05.02 18:00	vizsga	2761	2761
2011.06.07 18:00	tárgy	6095	6095
2011.11.28 18:00	vizsga	4897	4897
2012.01.16 18:00	tárgy	8120	5328
2012.01.30 16:00	tárgy		
2012.05.02 18:00	vizsga		

Richtig konfigurierte Server, richtige Leistungsdimensionierung!

Neptun – Anmeldung für LVAs

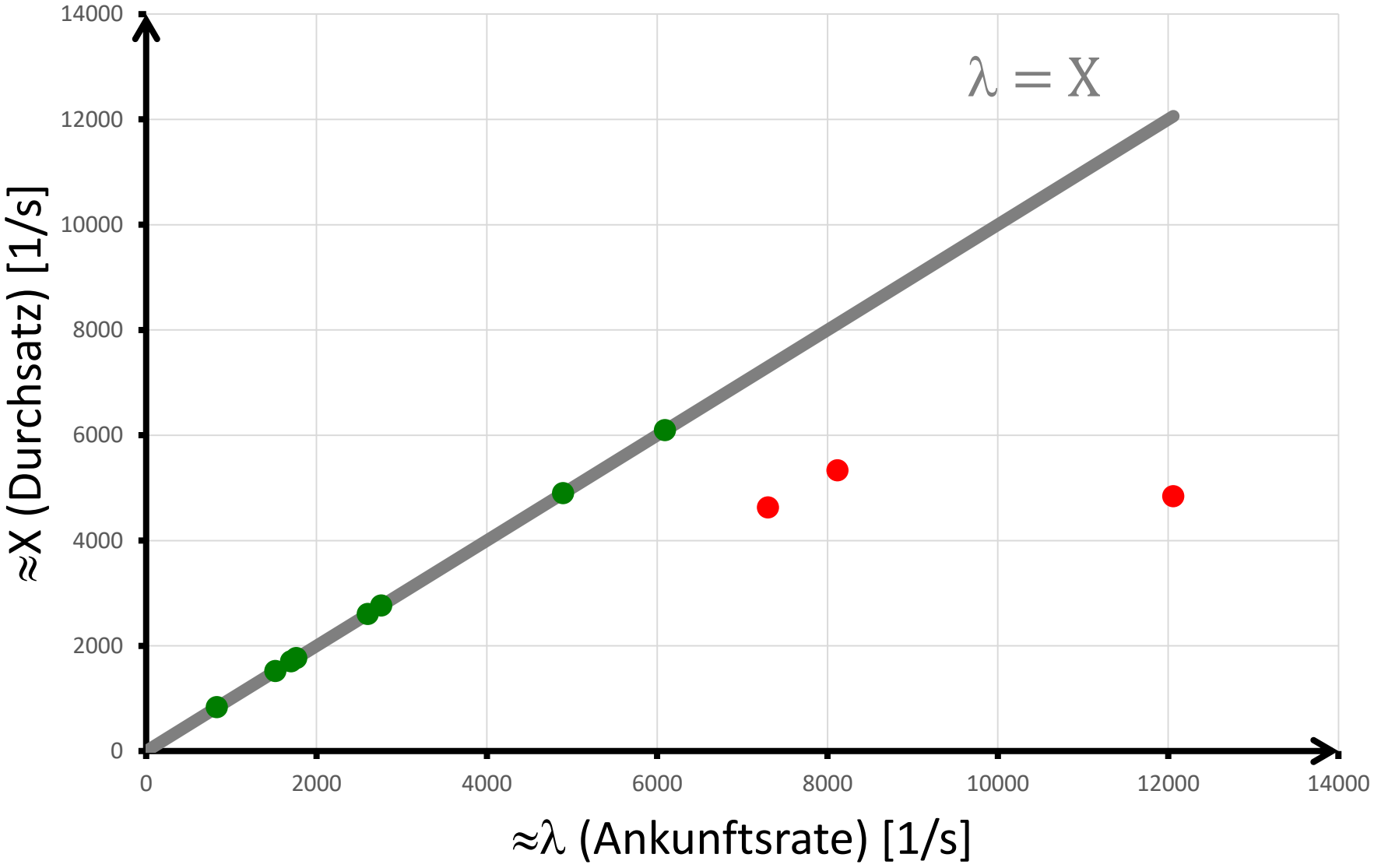
Dieselbe dient als Basis für die Verteidigung gegen (D)DoS

Das Ziel der Bestandeinschränkung ist es, das System in stabilem Zustand zu halten.

Dátum	Jelleg		
2010.11.29			1519
2010.12.22 06:0			2761
2011.01.10 18:0			6095
2011.01			4897
2		8120	5328
20			
201			
2012.			

Richtig konfigurierte Server, richtige Leistungsdimensionierung!

Vorblick: Datenvisualisation



RESSOURCEMODELLIERUNG

- Woher kommt die Grenze des Durchsatzes

Beispiel – Bewertung der Klausuren

- Die Bewertung einer Klausurarbeit dauert **15 Minuten**
- Wie viele Arbeiten **pro Stunde** werden **von einem** Lehrer abgefertigt?

$$X_{(1)}^{max} = \frac{1\ Klausur}{15\ Minuten} = 4\frac{Kl.}{h}$$

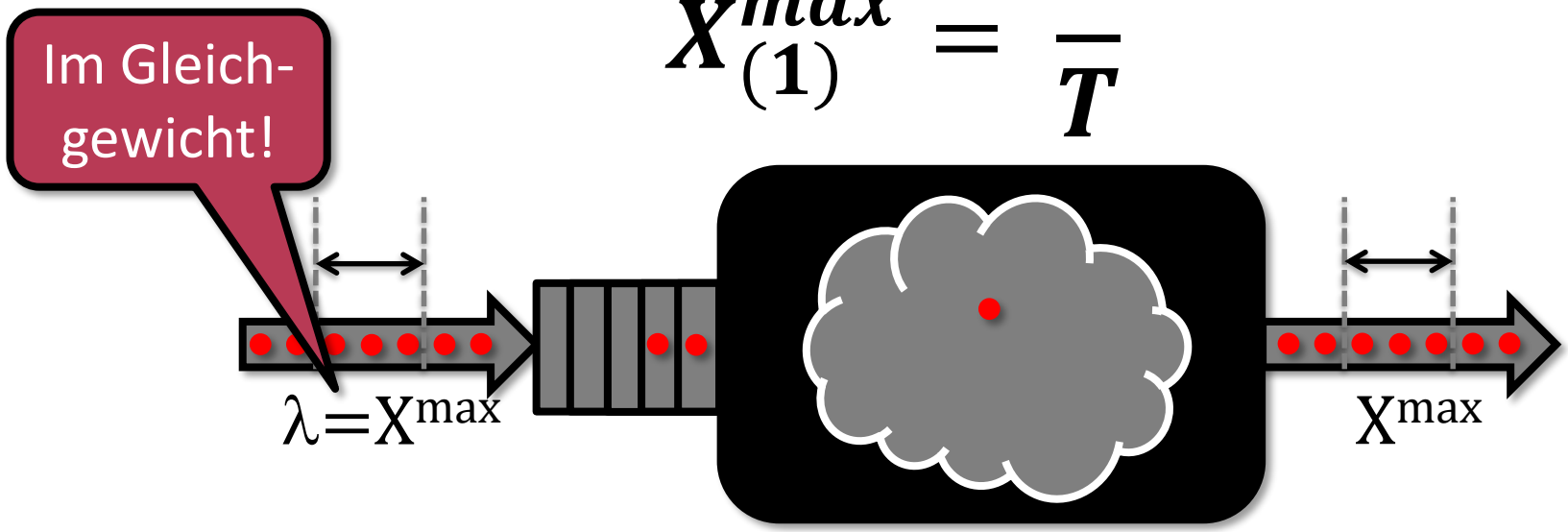
- Und **von acht** Lehrern?

$$X_{(8)}^{max} = 8 \times X_{(1)}^{max} = 8 \times \frac{1 Klausur}{15\ Minuten} = 32\frac{Kl.}{h}$$

Grenzdurchsatz eines Ein-Server-Systems

- Abfertigung von **höchstens einer** Anfrage zu gleicher Zeit
  - z.B. abgefertigt von einem Server, oder mit Schreiben einer gemeinsamen Variable
  - die anderen Anfragen/Prozessinstanzen **stehen Schlange**
- Dann bei $\mathbf{T}$ **Durchschnittsabfertigungszeit**:

$$X_{(1)}^{max} = \frac{1}{T}$$



Grenzdurchsatz eines Ein-Server-Systems

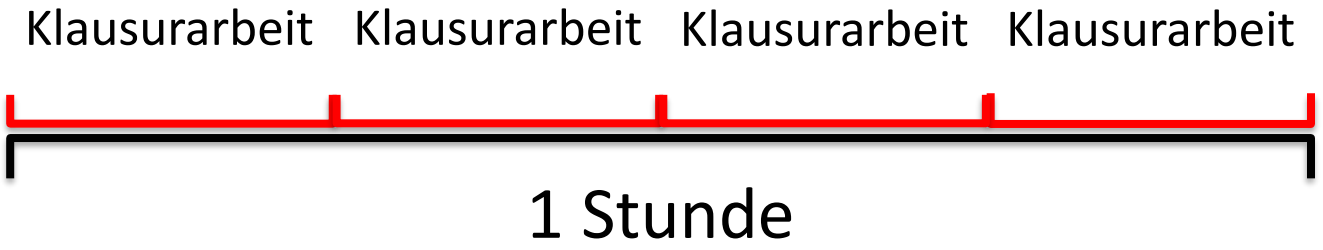
- Abfertigung von **höchstens einer** Anfrage zu gleicher Zeit
  - z.B. abgefertigt von einem Server, oder mit Schreiben einer gemeinsamen Variable
  - die anderen Anfragen/Prozessinstanzen **stehen Schlange**
- Dann bei $\mathbf{T}$ **Durchschnittsabfertigungszeit**:

$$X_{(1)}^{max} = \frac{1}{T}$$

„Wie viele Abfertigungen passen in einer Zeiteinheit?"

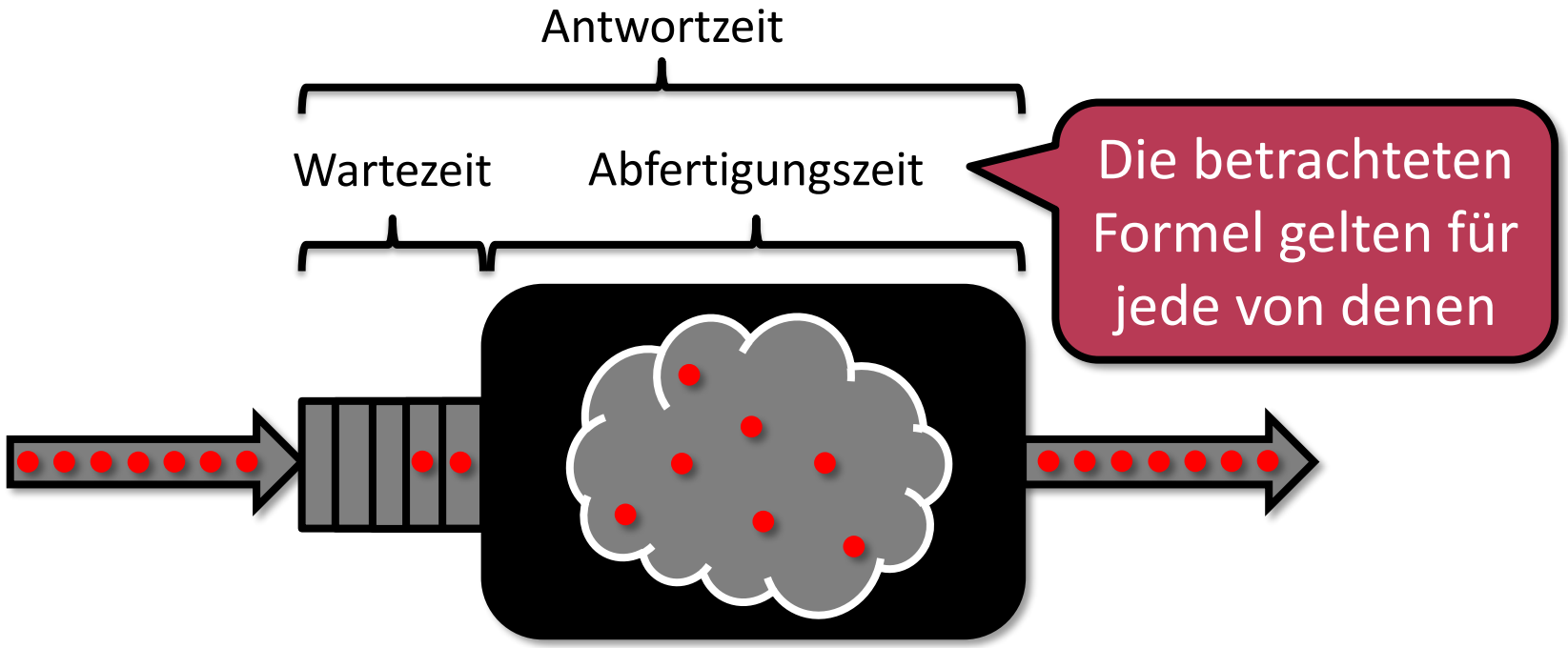
$T_{Kl.} = 15$ Min.

$X_{max} = 4\frac{1}{h}$



Messbare Zeiten

- **Wartezeit:** Warten auf die Ressource
- **Abfertigungszeit:** Abfertigung der Anfrage
- **Antwortzeit:** Wartezeit + Abfertigungszeit



Auslastung eines Ein-Server-Systems

- Auslastung: „Verhältnis des aktuellen und des Grenzdurchsatzes”

$$X_{(1)}^{max} = \frac{1}{T} \quad \Rightarrow \quad X_{(1)}^{max} \times T = 1$$

Auslastung eines Ein-Server-Systems

- Auslastung: „Verhältnis des aktuellen und des Grenzdurchsatzes”

$$X_{(1)}^{max} = \frac{1}{T} \quad \Rightarrow \quad X_{(1)}^{max} \times T = 1 = U^{max}$$

- Die Formel der Auslastung:

$$U = \frac{X}{X_{(1)}^{max}} = \frac{X \times T}{X_{(1)}^{max} \times T} = \frac{X \times T}{1} = X \times T$$

- Intuitiv:

„Wie grosser Teil der Zeiteinheit wird gebraucht, um die in einer Zeiteinheit ankommenden Anfragen mit T durchschnittlicher Bearbeitungszeit abzufertigen?”

Auslastung eines Ein-Server-Systems

- Auslastung: „Verhältnis des aktuellen und des Grenzdurchsatzes”

$$X_{(1)}^{max} = \frac{1}{T} \quad \Rightarrow \quad X_{(1)}^{max} \times T = 1 = U^{max}$$

- Die Formel der Auslastung:

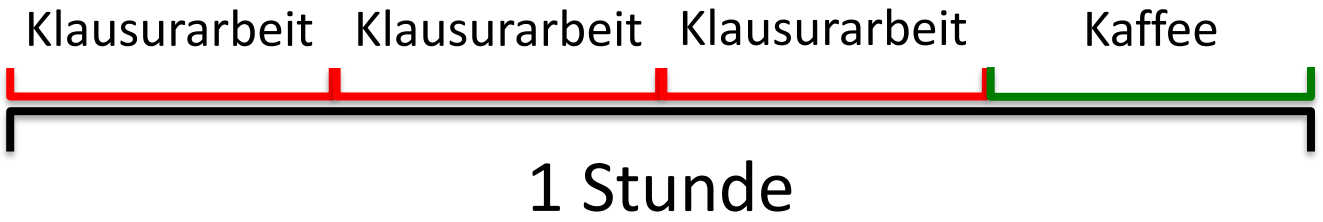
$$U = \frac{X}{X_{(1)}^{max}} = \frac{X \times T}{X_{(1)}^{max} \times T} = \frac{X \times T}{1} = X \times T$$

- Intuitiv:

$X = 3\,\frac{1}{h}$

$T_{Kl.} = 15$ Min.

$U = 75\%$

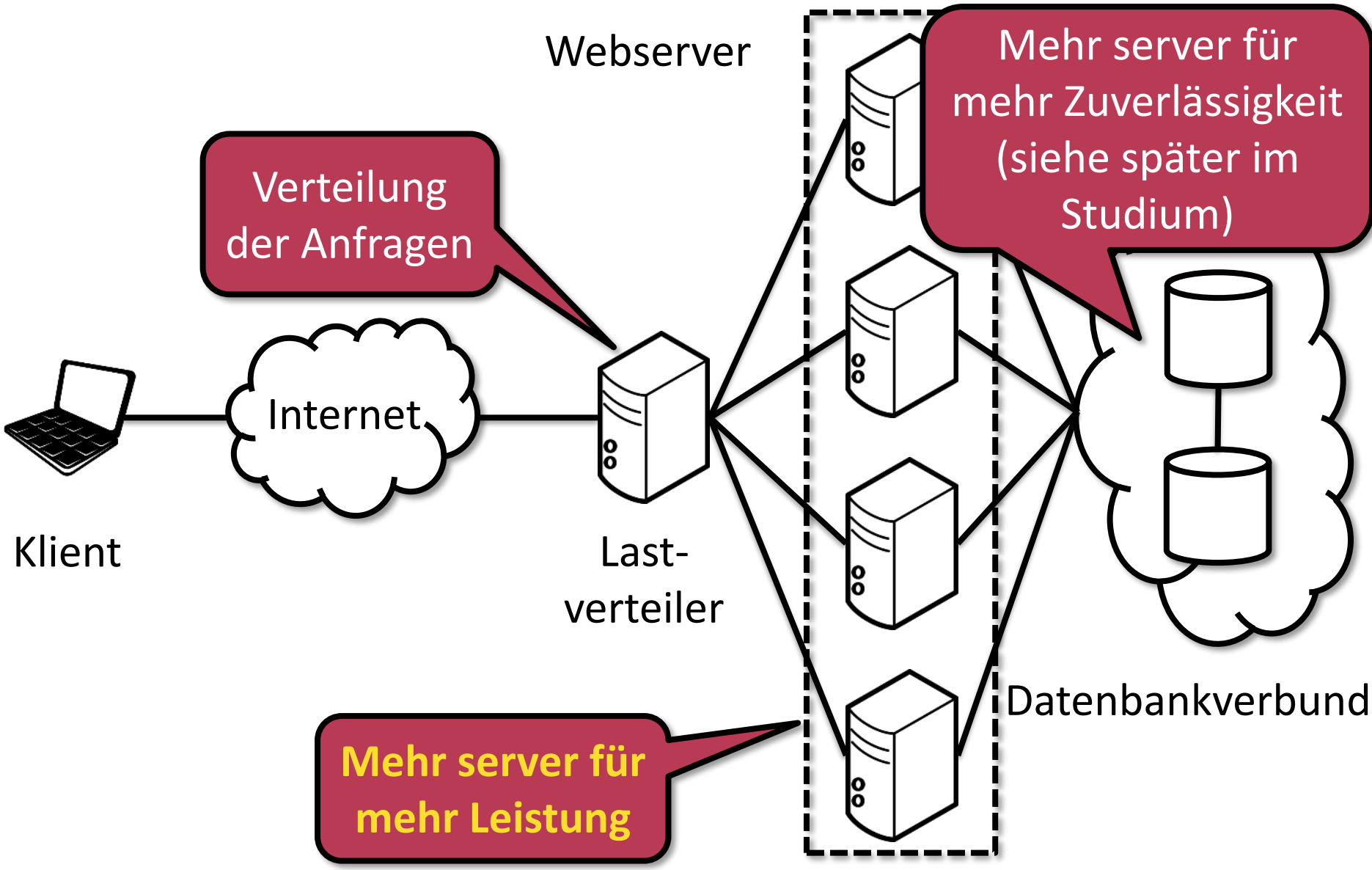


Ausschau: Skalierung

- **Vertikale Skalierung** (Scale-up):
  - Die **Leistung** der Bedieneinheiten wird erhöht
  - z.B. stärkeres CPU, mehr RAM
  - Einfach und herrlich ☺
  - Technologisch beschränkt ☹
- **Horizontale Skalierung** (Scale-out):
  - Die **Anzahl** der Bedieneinheiten wird erhöht
  - z.B. mehr CPU(-Kerne), mehr server
  - Theoretisch unbeschränkt fortsetzbar ☺
  - Zusätzliche Komplexität ☹

Scale-out im Alltag

Scale-out im Neptun



Grenzdurchsatz eines Mehr-Server-Systems

- Abfertigung von **höchstens K** Anfragen zu gleicher Zeit
  - z.B. abgefertigt von einem Serververbund von K Servern
  - die anderen Anfragen/Prozessinstanzen **stehen Schlange**
- Dann bei **T Durchschnittsabfertigungszeit**:

$$X_{(K)}^{max} = K \times X_{(1)}^{max} = \frac{K}{T}$$

Grenzdurchsatz eines Mehr-Server-Systems

- Abfertigung von **höchstens K** Anfragen zu gleicher Zeit
  - z.B. abgefertigt von einem Serververbund von K Servern
  - die anderen Anfragen/Prozessinstanzen **stehen Schlange**
- Dann bei **T Durchschnittsabfertigungszeit**:

Mit weiteren Serverinstanzen kann das System **skaliert** werden.

$$X_{(K)}^{max} = K \times X_{(1)}^{max} = \frac{K}{T}$$

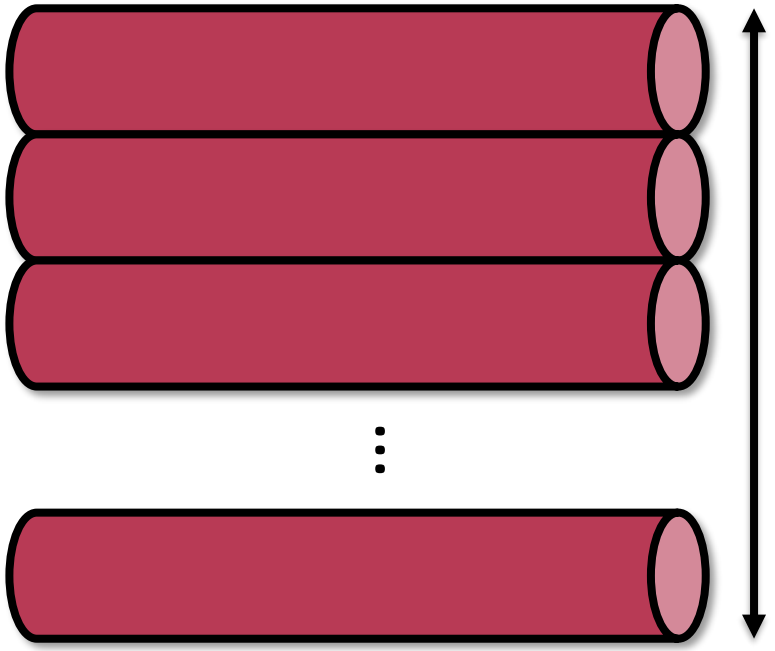
Analogie mit den Wasserröhren

Einzige exclusive Ressourceninstanz

Grenzdurchsatz:

$X_{(1)}^{max}$

K frei wählbare Ressourceninstanzen



Grenzdurchsatz :

$X_{(K)}^{max} = \mathrm{K} \times X_{(1)}^{max}$

Auslastung eines Mehr-Server-Systems

- Analog zu dem vorigen Gedankengang:

$$X_{(K)}^{max} = \frac{K}{T} \quad \Rightarrow \quad X_{(K)}^{max} \times T = K \quad ?$$

Auslastung eines Mehr-Server-Systems

- Analog zu dem vorigen Gedankengang:

$$X_{(K)}^{max} = \frac{K}{T} \quad \Rightarrow \quad X_{(K)}^{max} \times T = K = U_{(K)}^{max}$$

- Die Formel der Auslastung in diesem Fall:

$$U = \frac{X_{(1)}}{K} \times T$$

- Intuitiv:

„In wie großem Teil von K Zeiteinheiten würde ein einziger Server arbeiten?"

„Wie groß ist die Auslastung verglichen mit einem Ein-Server-System?"

Auslastung eines Mehr-Server-Systems

- Analog zu dem vorigen Gedankengang:

$$X_{(K)}^{max} = \frac{K}{T} \quad \Rightarrow \quad X_{(K)}^{max} \times \mathbf{T} = \mathbf{K}$$

- Die Formel der Auslastung in diesem Fall:

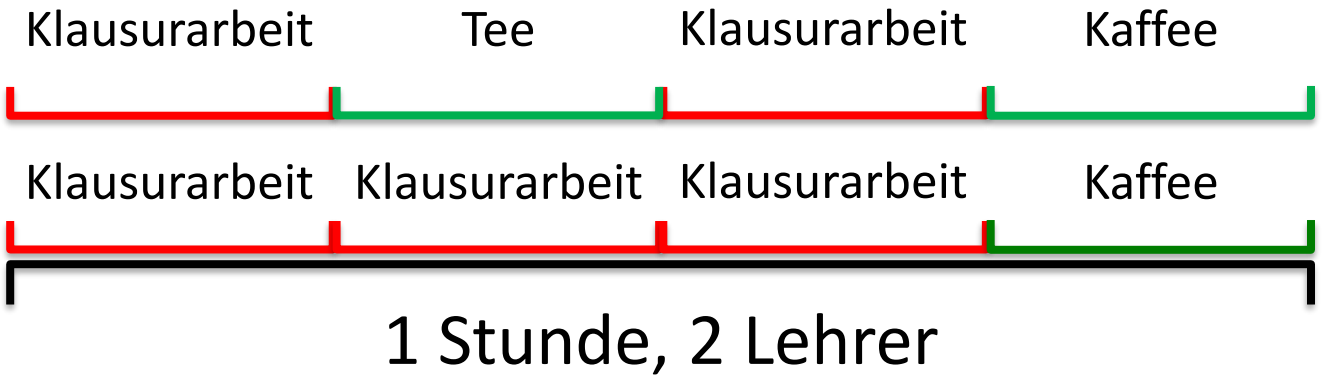
$$\mathbf{U} = \frac{X}{K} \times \mathbf{T}$$

- Intuitiv:

$X = 5\,\frac{1}{h}$

$T_{Kl.} = 15$ Min.

$U = 62{,}5\%$



Zusammenfassung

- **Stabiler Zustand:**
  - kann mit Durchschnittswerten gerechnet werden
  - $\lambda = X$ (Ankuftsrate = Durchsatz)
- **Grenzdurchsatz:**
  - der grösste erreichbare Durchsatz
  - $X^{max} = \frac{K}{T}$ (bei K frei wählbaren Ressourceninstanzen)
- **Auslastung**:
  - Verhältnis des aktuellen und des Grenzdurchsatzes
  - $U = \frac{X}{K} \times T$ (bei K frei wählbaren Ressourceninstanzen)

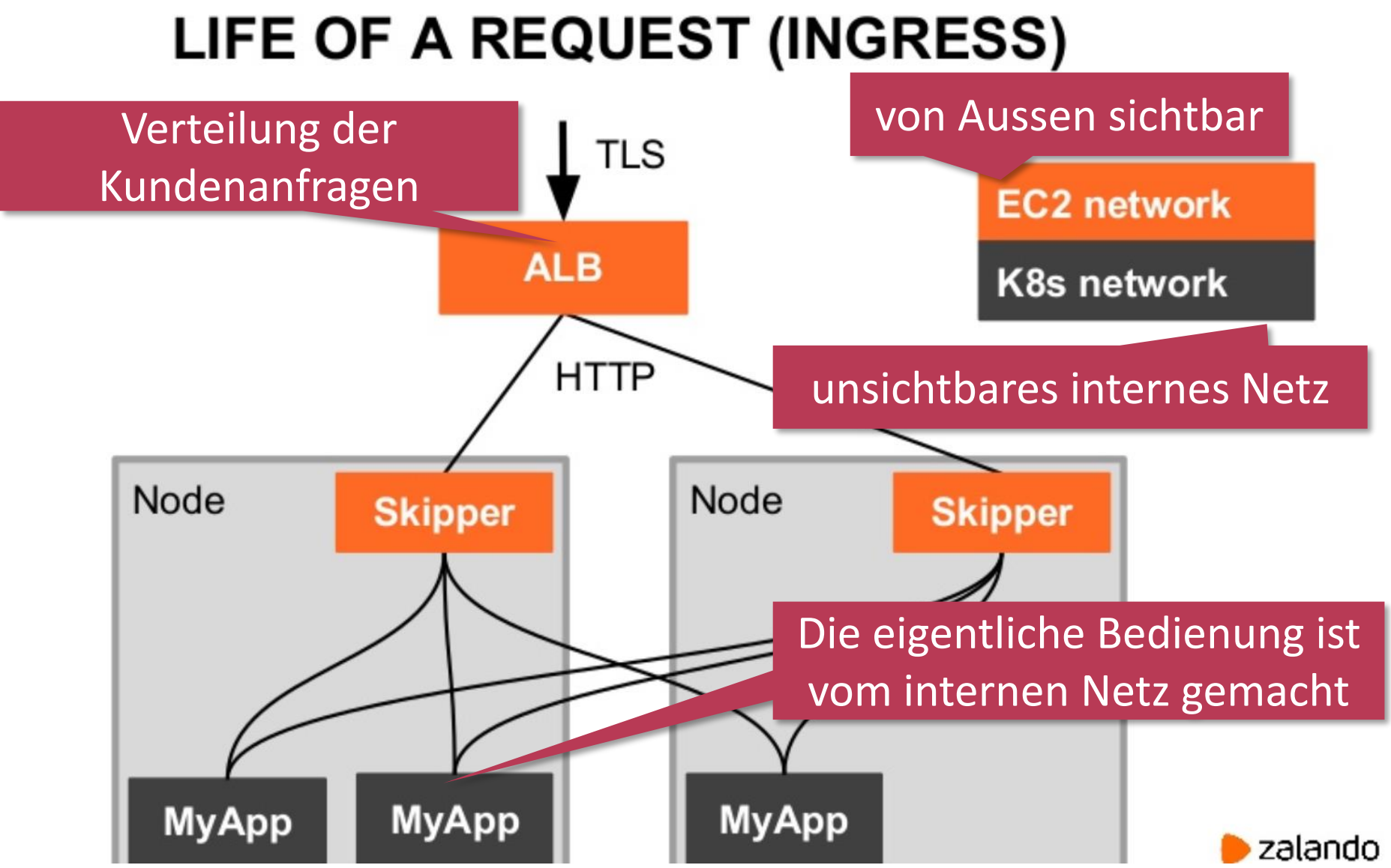
BEISPIEL

Warum konnte man von diesen feinen Socken beim Black Friday keine kaufen?

(oder wie die Skalierung zurückschlagen kann …)

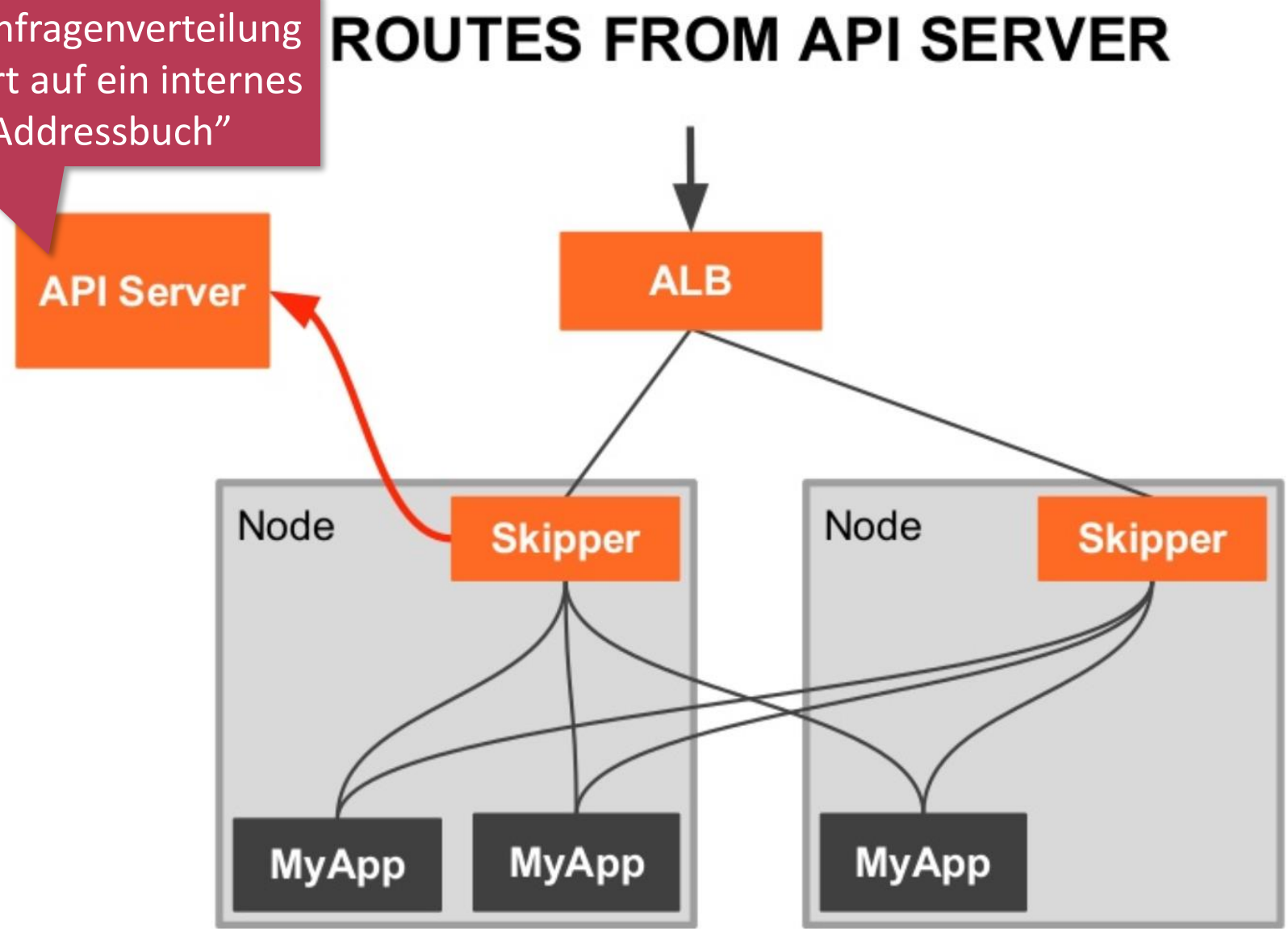
Henning Jacobs: Running Kubernetes in Production: A Million Ways to Crash Your Cluster - DevOpsCon Munich 2018
https://www.slideshare.net/try_except_/running-kubernetes-in-production-a-million-ways-to-crash-your-cluster-devopscon-munich-2018
https://github.com/zalando-incubator/kubernetes-on-aws/blob/dev/docs/postmortems/jan-2019-dns-outage.md

Skalierung in der Praxis

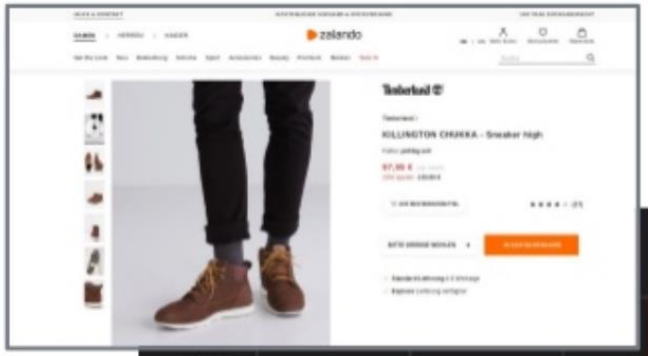


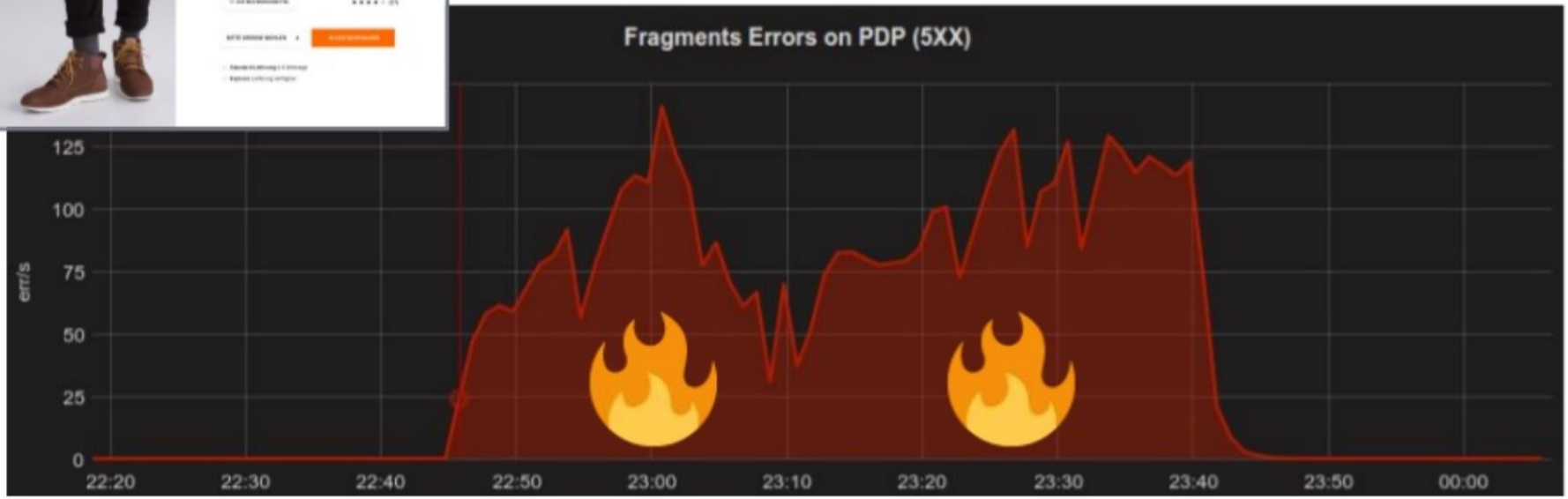
Wo reisen die Anfragen entlang?

Die Anfragenverteilung basiert auf ein internes „Addressbuch”

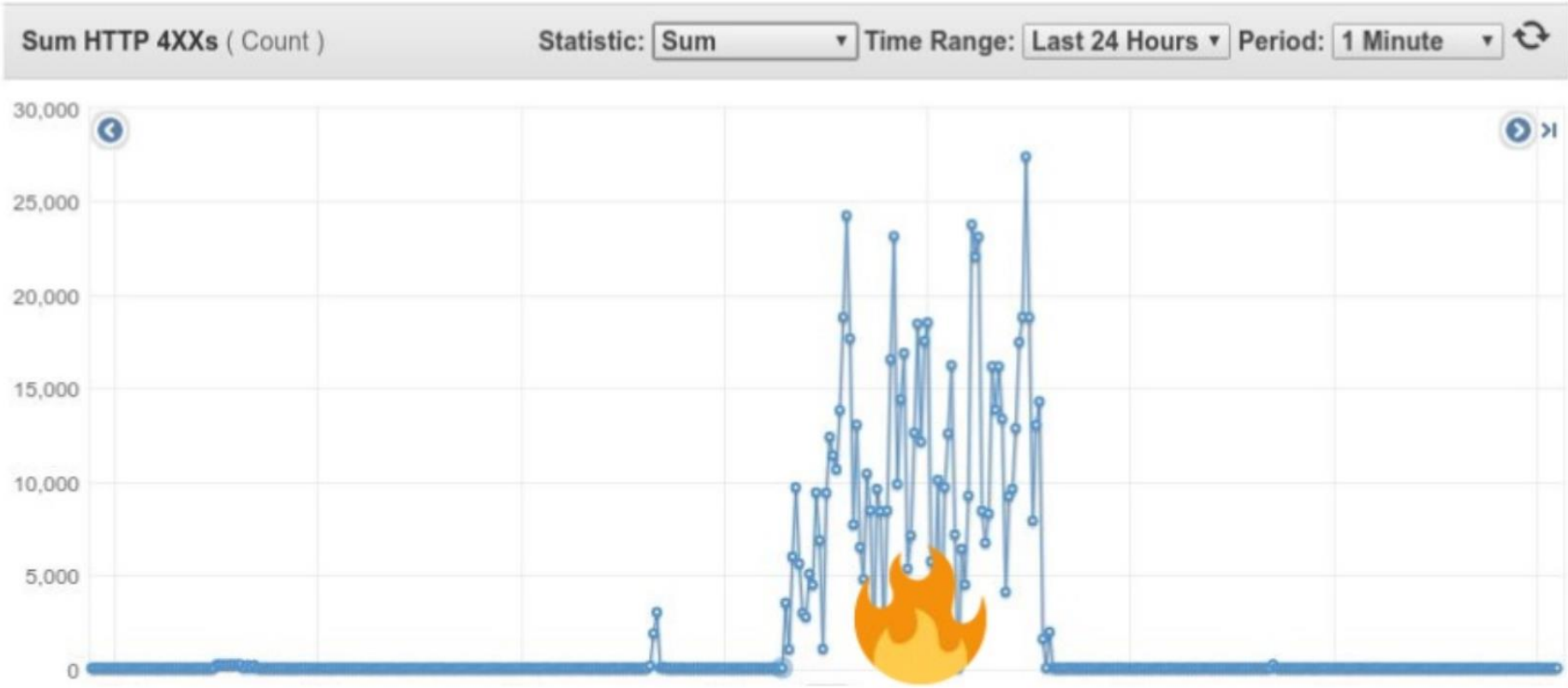


Die von Aussen sichtbaren Fehler



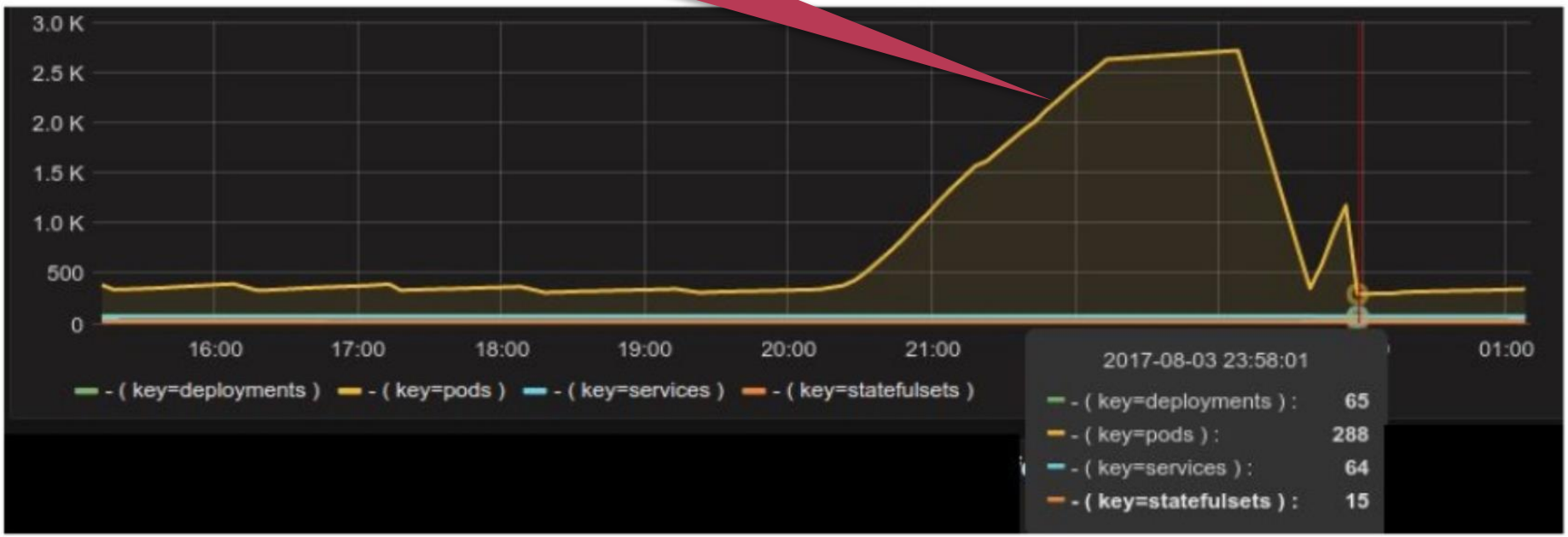


404 Errors (falsche Addresse)

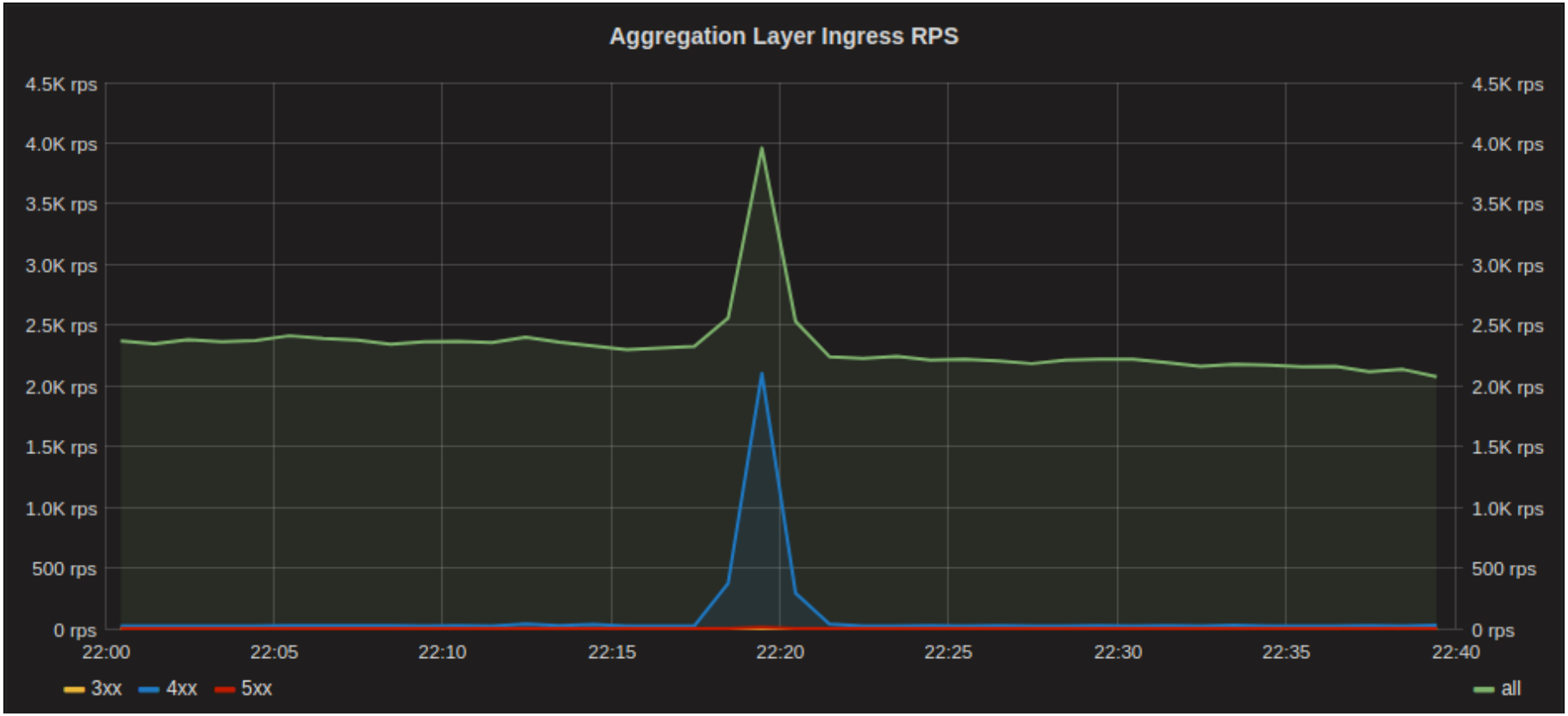


Die Anzahl der elementaren Server (Pods)

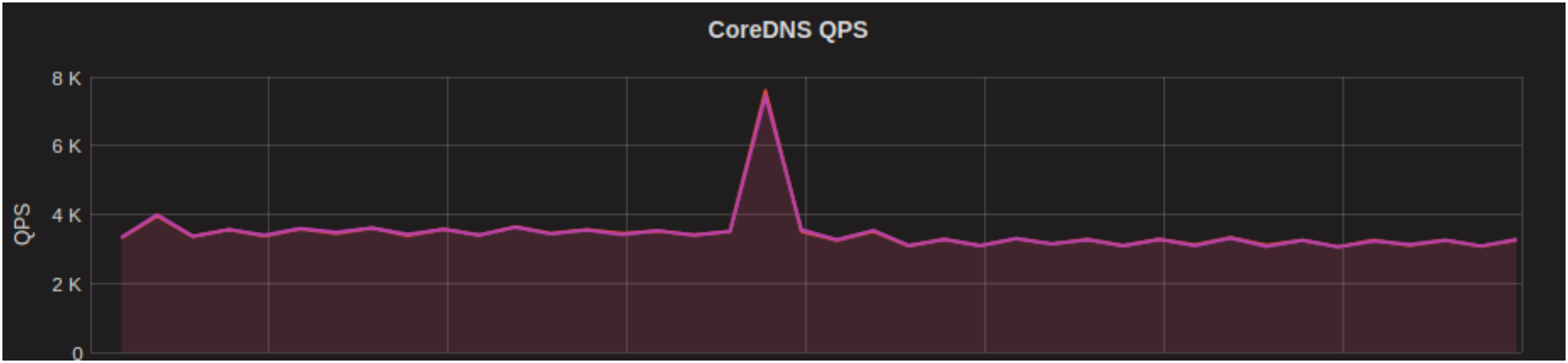
Mehr Ressourcen
→ mehr Fehler



Anwendungsfehler in letzten 30 Minuten



Addressabfragen in den letzten 30 Minuten



Die Lehre aus der Geschicht’

- Mehr Ressourcen benötigen mehr Operationen
  - Hier z.B. mehr Addressabfragen (DNS)
- Addressabfragen von kleinem/einfachem Ressource bedient
- Ressource gesättigt
  - **Die Zustandsmaschine der Überwachung ist falsch**
  - Richtig: „Sättigend” Zustand != „Neu startend” Zustand
  - Unnötige neu Starts, unnötige neue Addressen
    - → unnötige Nachrichten/Aktionen
    - → Überwachung nach einer Zeit auch zusammengebrochen
    - → kein Ressourcemodell kann es vorhersagen …